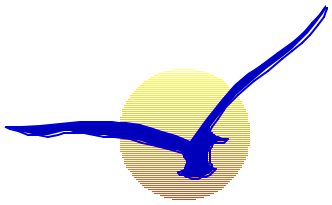
01175.21-01

Free Flight Simulation Infrastructure

Fiscal Year 2000 Final Report

D. R. Schleicher, P. C. Davis, E. Wallace, S. Shah, S. Dorsky,
T. Mueller, E. Jones, J. Krozel, G. J. Couluris, D. Dow



Seagull Technology, Inc.

Prepared for:

National Aeronautics and Space Administration
Ames Research Center
Moffett Field, CA 94035-1000

Under Subcontract to:

Honeywell Inc.
Houston Engineering Center
Houston, TX 77058

Honeywell Subcontract No. 8021308, Line Item 8
NASA Prime Contract No. NAS2-98001

Seagull Project No. C175.21

30 March 2001

Acknowledgments

The development of the Fiscal Year 2000 Free Flight Simulation (FFSim) Infrastructure was performed for the Advanced Air Transportation Technologies (AATT) program of the National Aeronautics and Space Administration (NASA). Mr. Jeff Maddalon, Mr. Richard Barhydt, Ms. Sheila Conway, Mr. Mike Palmer, Mr. David Wing, and Mr. Mark Ballin of NASA Langley Research Center, provided valuable technical and programmatic support.

The Fiscal Year 2000 FFSim Infrastructure software development effort was conducted by Seagull Technology, Inc. Mr. David Schleicher served as principal investigator and managed the completion of the project. Mr. Paul Davis served as the lead software engineer and managed the software development and integration effort associated with this project. Ms. Erin Wallace, Ms. Shreya Shah, Ms. Susan Dorsky, and Mr. Darren Dow comprised the remainder of the software team and each contributed in numerous ways to this effort. Mr. David Schleicher was supported in the simulation systems engineering tasks by Mr. Tysen Mueller, who developed the Secondary Surveillance Radar, Mode S and VDL-4 datalink communication models, and by Mr. Ed Jones, Dr. Jimmy Krozel, and Dr. George Couluris, who worked on the ADS-B, FIS-B, and Host Computer System Emulator specifications, respectively. Additionally, Dr. Krozel and Ms. Dorsky provided the AOP functional specification review and the architecture recommendations.

The authors also wish to thank Mr. Bill Corwin, who as our point-of-contact to the prime contractor, Honeywell, provided valuable leadership and insight.

Abstract

Over the past twenty years, unprecedented demand for air travel has outpaced the design of the National Airspace System (NAS). Insufficient capacity, limited access, and excessive operating restrictions have led to significant increases in user costs and delays, and an overall decrease in efficiency for all users [RTCA95]. Furthermore, the market demand for air travel is expected to increase several-fold in the coming decade.

In an attempt to redesign the NAS to accommodate the expected rise in demand in air transportation, the Advanced Air Transportation Technologies (AATT) program of the National Aerospace and Space Administration (NASA) is working to realize a capability known broadly as “Free Flight”. The transition from the existing air-traffic-management system to a pragmatic realization of Free Flight will involve significant developments in technology and procedures, both for air-traffic-control systems on the ground and in aircraft flight-deck systems.

NASA Langley Research Center (LaRC) is undertaking the development of a Free-Flight Simulation (FFSim) for distributed air-ground research. FFSim is a distributed collection of desktop simulation components that provide advanced and medium fidelity aircraft models, communication, navigation, surveillance, and air traffic management components of the future NAS. These components enable multiple researchers and other human subjects to participate in a real-time simulation of future air traffic control operations. FFSim is intended to help both NASA’s AATT program and US ATM research and development efforts. FFSim is intended to: 1) allow researchers to investigate the feasibility of new air-traffic-management operational concepts, 2) provide an environment for future cockpit and ATM automation design, and 3) provide human-in-the-loop demonstrations of future ATM concepts.

This report documents the software development and engineering effort to realize Phase 1R of the FFSim infrastructure. The Phase 1R FFSim infrastructure contains communication, surveillance, and air traffic management components that have been designed to operate in conjunction with NASA-LaRC’s Aircraft Simulator for Traffic Operations Research (ASTOR) and NLR’s Traffic and Experiment Manager (TEM). The Phase 1R FFSim infrastructure allows the modeling of current and future NAS infrastructure including: air-to-air and air-to-ground surveillance information broadcast through Automatic Dependent Surveillance-Broadcast (ADS-B), addressable air-ground communication through Controller-Pilot Data Link Communication (CPDLC), ground-to-air flight information through Flight Information Services-Broadcast (FIS-B), an air traffic controller workstation based on an integration of the Center-TRACON Automation System (CTAS) through a Host Computer System Emulator (HCSE), Mode S and VDL-4 air-ground datalink communication models, and Secondary Surveillance Radar (SSR).

Table of Contents

Symbols and Abbreviations

AATT	Advanced Air Transportation Technologies program
ADRS	Aeronautical Datalink and Radar Simulator
ADS-B	Automatic Dependent Surveillance - Broadcast
AOC	Airline Operational Center
AOL	Airborne Operations Lab at NASA ARC
AOP	Autonomous Operations Planner
ARC	Ames Research Center
ARTCC	Air Route Traffic Control Center
ARTS	Automated Radar Terminal System
ASTOR	Aircraft Simulator for Traffic Operations Research
ATC	Air Traffic Control
ATCBI	Air Traffic Control Beacon Interrogator
ATCDST	Air Traffic Control Decision Support Tool
ATCRBS	Air Traffic Control Radar Beacon System
ATM	Air Traffic Management
ATMFS	Air Traffic Management Facility Simulation
ATSP	Air Traffic Service Provider
CD&R	Conflict Detection & Resolution
CDTI	Cockpit Display of Traffic Information
CDU	Control Display Unit
CID	Computer ID
CM	Communications Manager CTAS software process
CNS	Communications, Navigation, and Surveillance
CORBA	Common Object Request Broker Architecture
CPDLC	Controller-Pilot Data Link Communications
CTAS	Center-TRACON Automation System
DFW	Dallas / Fort Worth Airport
DST	Decision Support Tool
ETA	Estimated Time of Arrival
FastWin	FMS/Auto-flight Simulation Tools for Windows, NASA-developed FMS-modeling software
FFSim	Free Flight Simulation

FIS-B	Flight Information Services - Broadcast
FMS	Flight Management System
GFSK	Gaussian Frequency Shift Keying
GPS	Global Positioning System
GRDT	Ground Requirements Development Tool
GSC	General Signaling Channel
GUI	Graphical User Interface
HCS	Host Computer System
HCSE	Host Computer System Emulator
ICAO	International Civil Aviation Organization
ID	Identification
IPC	Inter-process Communication
ISM	Input Source Manager CTAS software process
LAAS	Local-Area Augmentation System to the Global Positioning System
LaRC	Langley Research Center
MCP	Mode Control Panel
METAR	Meteorological Terminal Aerodrome Forecast
MIL-STD	Military Standard
MOA	Military Operating Area
NAS	National Airspace System
NASA	National Aeronautics and Space Administration
ND	Navigation Display
NEAN	North European ADS-B Network
NLR	Nationaal Lucht en Ruimtevaart Laboratory (The Netherlands National Aerospace Laboratory)
NRFMS	NASA Research Flight Management System
OOD	Object-Oriented Design
PAS	Pseudo Aircraft Systems
PGUI	CTAS' Plan-view Graphical User Interface Display
PSR	Primary Surveillance Radar
PVD	Plan-view Display
RTA	Required Time of Arrival
RTCA	formerly the Radio Technical Commission for Aeronautics

RTO	Research Task Order
SCC	Simulation Command and Control
SIGMET	Significant Meteorological Information
SSR	Secondary Surveillance Radar
SUA	Special Use Airspace
TCAS	Traffic Alert and Collision Avoidance System
TCP	Trajectory Change Point
TCP/IP	Transmission Control Protocol/Internet Protocol
TEM	Traffic and Experiment Manager
TIS-B	Traffic Information Services - Broadcast
TMA	Traffic Management Advisor
TRACON	Terminal Radar Approach Control
UAT	Universal Access Transceiver
UML	Unified Modeling Language
UTC	Coordinated Universal Time
VDL	Very High Frequency Digital Link
VDL-3	VDL Mode 3
VDL-4	VDL Mode 4
VHF	Very High Frequency
WAAS	Wide-Area Augmentation System to the Global Positioning System
ZFW	The Dallas / Fort Worth Air Route Traffic Control Center

1 Introduction

This report documents the software development and engineering effort to realize a free-flight simulation infrastructure for the Aircraft Systems and Operations sub-element of the National Aeronautics and Space Administration (NASA)'s Advanced Air Transportation Technologies (AATT) program. This introductory section provides a high-level overview of the continuing Free Flight Simulation (FFSim) Infrastructure development, its motivation, capabilities, and numerous components.

1.1 Motivation

In the past twenty years, increased demand for air travel has outpaced the design of the National Airspace System (NAS) [FAA98]. Insufficient capacity, limited access, and excessive operating restrictions have led to significant increases in user costs and delays, and an overall decrease in efficiency for all users [RTCA95]. Furthermore, the market demand for air travel is expected to increase several-fold in the coming decade. Recent technological advances now provide the opportunity to redesign the NAS to substantially increase capacity and efficiency, while maintaining, and possibly improving, safety.

In an attempt to redesign the NAS to accommodate the expected rise in demand to air transportation, a broad concept known as "Free Flight" has been proposed. The primary difference between today's direct-route-clearance approach and Free Flight will be the pilot's ability to operate the flight without being required to follow specific route, speed, and altitude clearances. The transition from the existing ground-based air traffic management system to Free Flight will involve significant developments in technology and procedures, both for air traffic control systems on the ground and aircraft flight-deck systems. The realization of Free Flight will ultimately be a distributed system, comprised of a vast number of individual technologies and procedures.

In recent years, a great deal of attention has been given to some of the individual components that will play a central role in Free Flight. For operations in air traffic control (ATC) facilities on the ground, researchers have been designing and field-testing the Center-TRACON Automation System (CTAS) [EDG93, DE95]. Commercial industry organizations have led the way in researching and developing airborne technology for the flight deck, such as advanced flight management system (FMS) avionics, conflict probes, and strategic flight planners.

In contrast to the individual ATC and airborne technologies, less attention has been given to date to the overall system implementation of free flight, and the essential underlying communication, navigation, and surveillance (CNS) components. It is the purpose of this work is to make a significant contribution to the air traffic management (ATM) research community by developing a research tool that models both current and envisioned CNS technologies and allows evaluation of Free Flight concepts in a systems context.

1.2 Capabilities

The primary objective of this effort is to develop a comprehensive, distributed simulation infrastructure to evaluate the efficacy of candidate free-flight technologies and concepts from an overall systems perspective. The FFSim infrastructure is designed and implemented to model a variety of technologies that are considered important to realize candidate near-term and future Free Flight concepts. The technologies include various communication, surveillance, and air traffic control automation components. Examples of such infrastructure components are Automatic Dependent Surveillance – Broadcast (ADS-B), Flight Information Services – Broadcast (FIS-B), Controller-Pilot Data Link Communications (CPDLC), Secondary Surveillance Radar (SSR), Mode S and VDL-4 air-ground datalinks, Wide-Area and Local-Area Augmentation Systems (WAAS and LAAS) to the Global Positioning System (GPS), and the Center-TRACON Automation System (CTAS).
.

2 FFSim Phase 1R Architecture

FFSim is a distributed simulation system comprised of many airborne, ATM, and CNS components. Figure 2-1 depicts some of these components at the conceptual level.

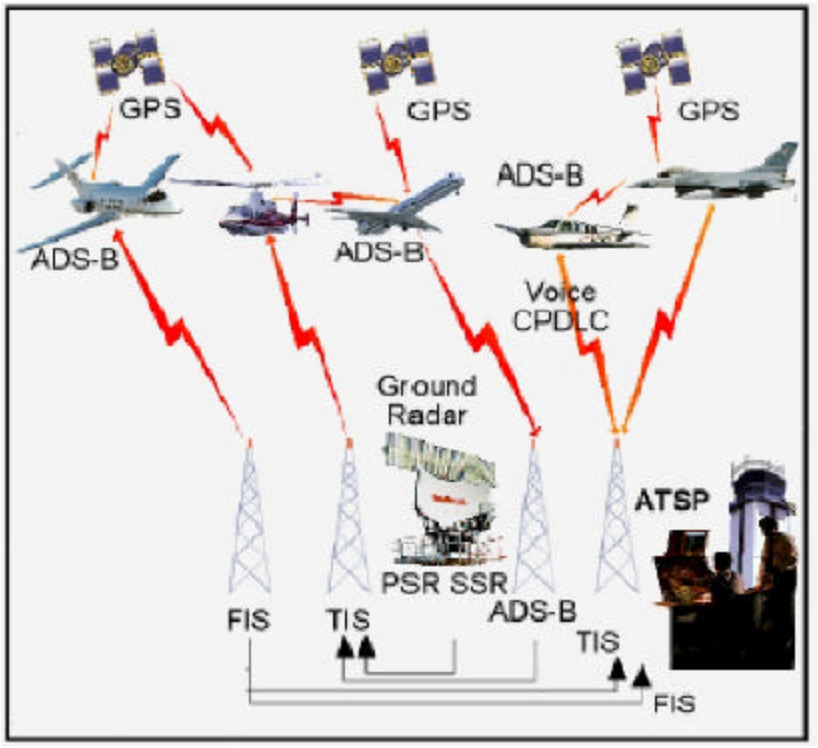


Figure 2-1 Free-Flight Simulation Elements

In Fiscal Year 1999 [DSD00], the System Architecture, Version 1.0 was developed and successfully demonstrated. A high-level view of the architecture is shown in Figure 2-2. The Simulation Infrastructure supported 3 Pilot Stations equipped with NASA-LaRC's FastWin [PAW97], NASA-ARC's CDTI [JBD97] with conflict detection and resolution, and CNS and simulation infrastructure including GPS, ADS-B and CPDLC. The Infrastructure also supported three PAS [PAS97] pseudo-pilot stations and two controller stations equipped with conflict detection and resolution, the CTAS PGUI [EDG93] and simulation and CNS infrastructure including ADS-B and CPDLC.

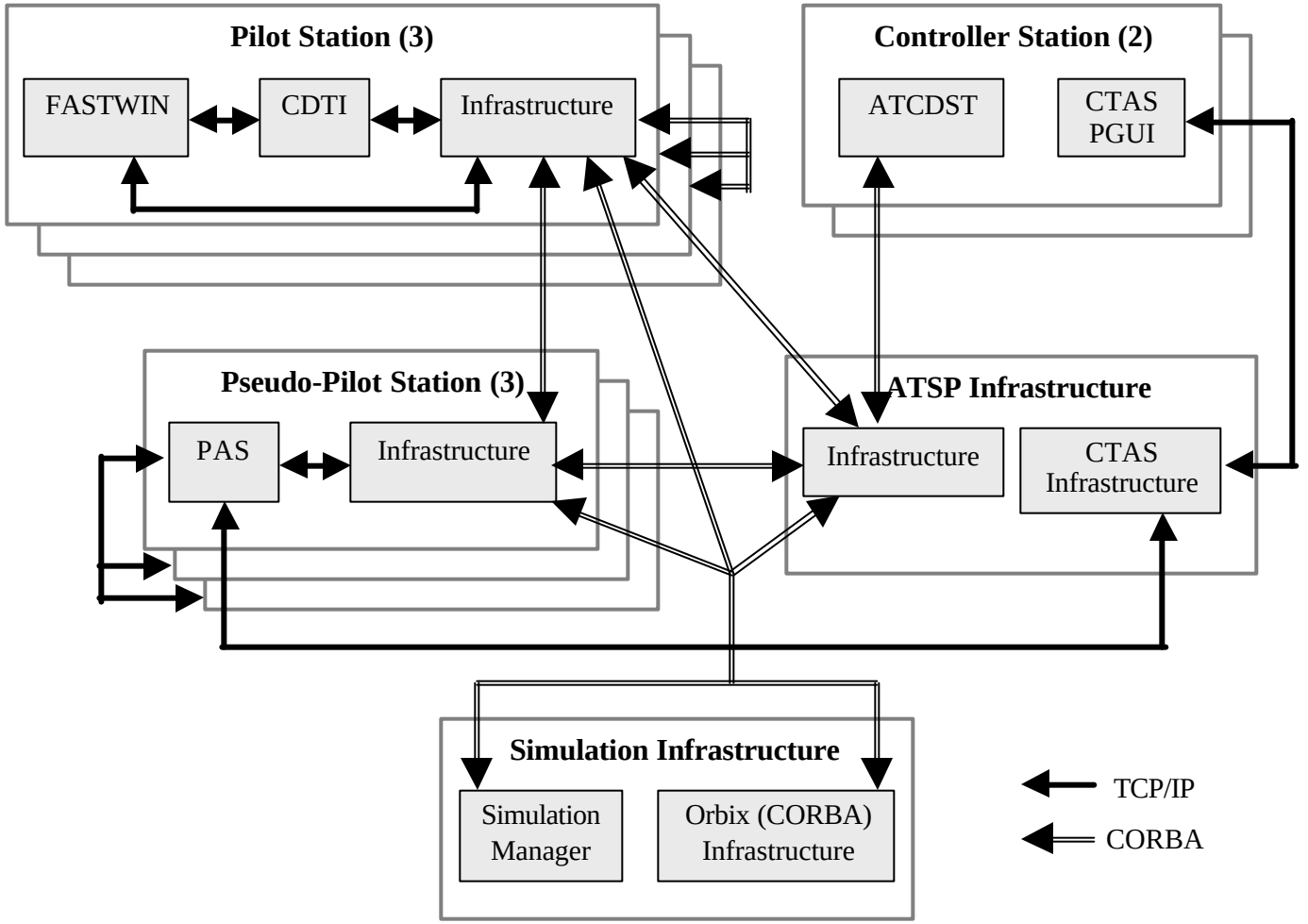


Figure 2-2 FFSim Phase 1 System Architecture [DSD00]

In Fiscal Year 2000, NASA decided to revise the Phase 1 Architecture for several reasons including: 1) facilitate integration of NLR's Traffic and Experiment Manager (TEM) for simulation command and control functions; 2) support the multiple-ASTOR pseudo-aircraft concept to avoid the need for multiple conflict detection and resolution (CD&R) models for both piloted and pseudo-piloted aircraft; and 3) eliminate Orbix, the proprietary CORBA implementation, which would have difficulty satisfying the inter-process communications performance requirements for the multiple-ASTOR concept. In February 2000, meetings were held at NASA Langley Research Center between representatives of NASA-LaRC, Seagull Technology, and NLR, to formulate a new FFSim architecture. The resulting re-architecture, known as Phase 1R, is shown in Figure 2-3.

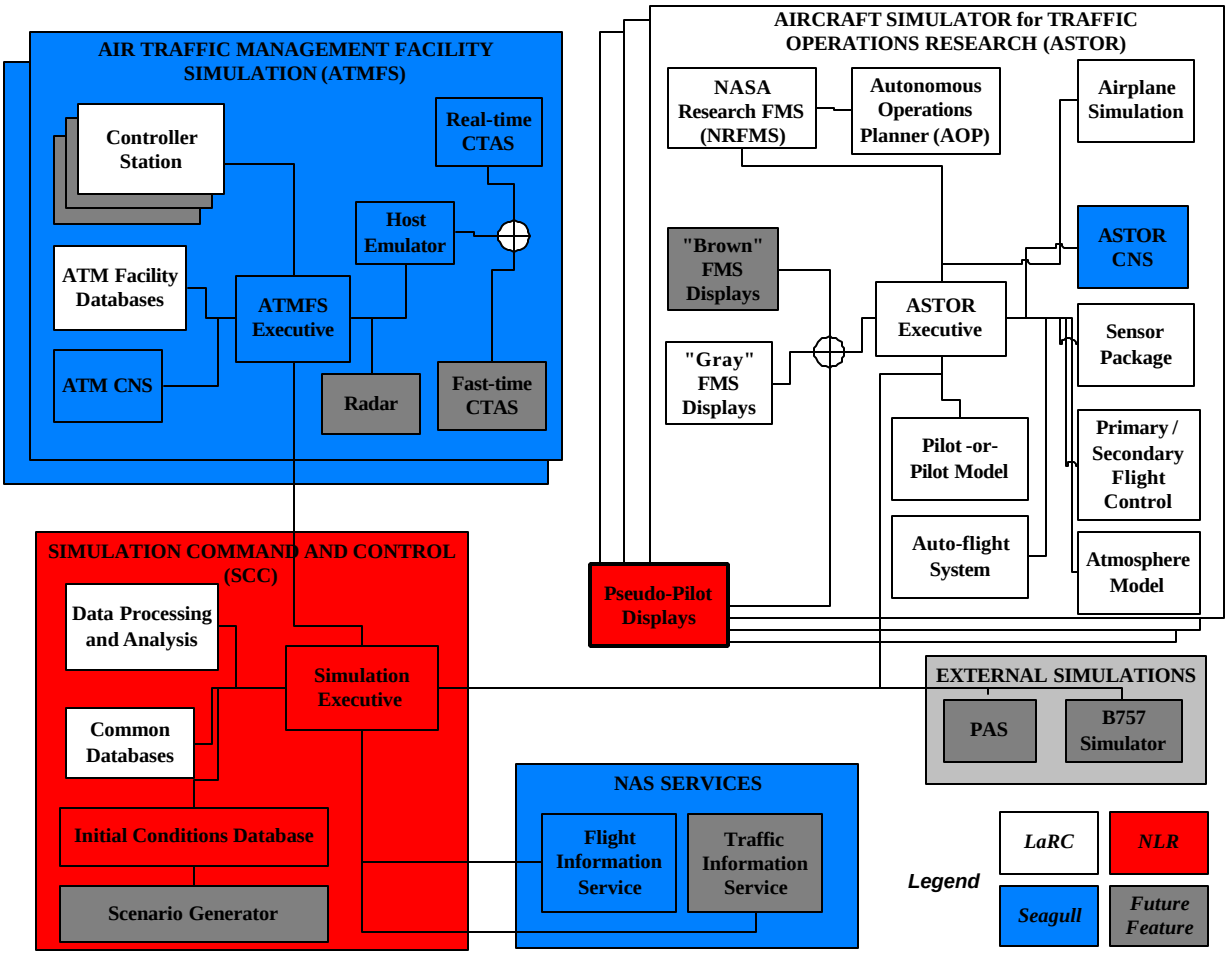


Figure 2-3 FFSim Phase 1R System Architecture

The responsibilities for the various components were divided between NLR, NASA-LaRC, and Seagull Technology. NLR was responsible for the Simulation Executive, the Pseudo-pilot displays and the Initial Conditions Database. NASA-LaRC was responsible for all of the ASTOR (except CNS and pseudo-pilot displays), the Controller Station, and the ATM Facility Databases. Seagull Technology was responsible for the ATM and ASTOR CNS, the ATMFS Executive, the Host Emulator, CTAS integration, and the FIS-B Service. Additionally, Seagull Technology was to develop functional specifications for Secondary Surveillance Radar application and the Datalink Communications Models that would reside in the Simulation Executive.

3 Phase 1R Infrastructure

For the RTO-21 Phase 1R Infrastructure effort, Seagull Technology developed requirements and functional specifications for the following:

- Automatic Dependent Surveillance -Broadcast (ADS-B)
- Air Traffic Management Facility Simulation (ATMFS) Executive
- Flight Information Services – Broadcast (FIS-B)
- Controller Pilot Datalink Communications (CPDLC)
- Host Computer System Emulator (HCSE)
- Mode S Datalink Communication Model
- VDL-4 Datalink Communication Model
- Secondary Surveillance Radar

Software was developed for the following components:

- ADS-B
- ATMFS Executive
- FIS-B
- CPDLC

Additionally, a review of the NASA-LaRC's AOP functional specifications was performed and a high-level AOP-FFSim integration software architecture was formulated.

What follows is a description of the work accomplished during the Phase 1R RTO-21 effort for each functional component and its software implementation, where appropriate.

3.1 Automatic Dependent Surveillance - Broadcast

The Automatic Dependent Surveillance - Broadcast (ADS-B) component was designed to integrate into the new FFSim Phase 1R architecture (shown in Figure 2-3). A detailed functional specification of the Phase 1R ADS-B model was developed and presented to NASA-LaRC at an August 2000 design review [SJD01]. Following its approval by NASA, a software design was produced and documented [D01-1]. Once the software design was reviewed and approved by NASA-LaRC, the ADS-B receiver and transmitter software was implemented in C++ and tested on both Windows NT and UNIX (Solaris) operating systems.

The following sections describe the key features of the ADS-B functional design and software.

3.1.1 Functional Description

The ADS-B model interacts with the other major FFSim components as shown in Figure 3-1.

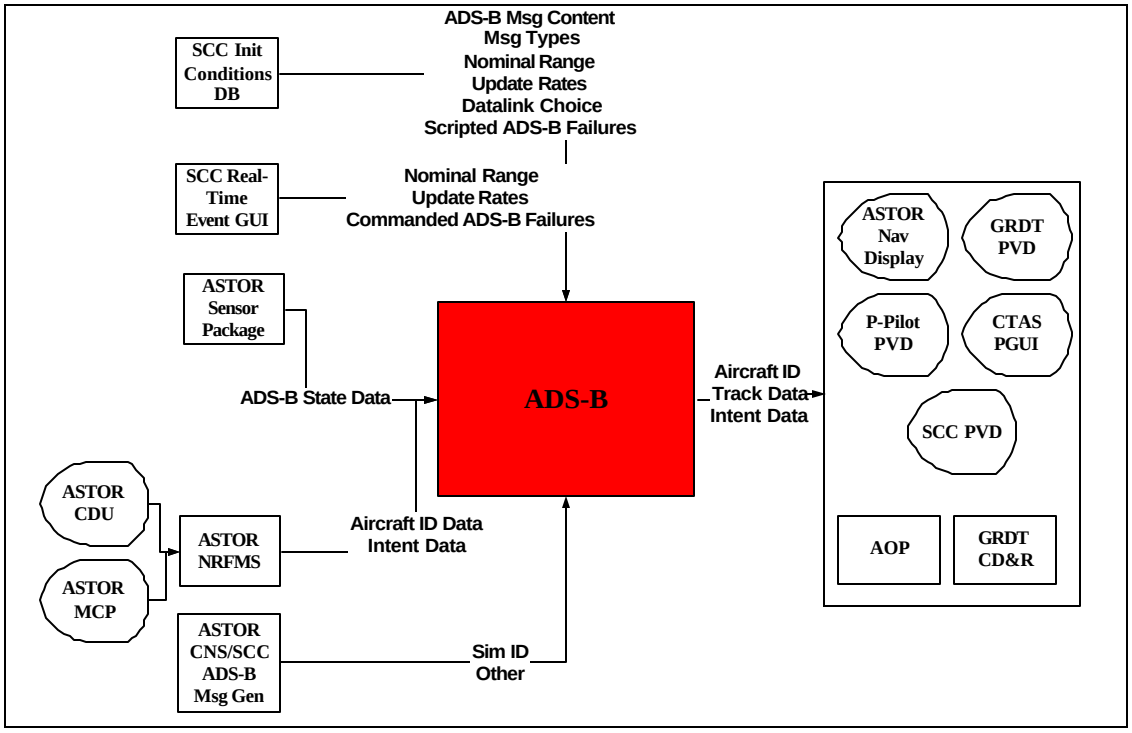


Figure 3-1 ADS-B Model Inputs and Outputs

Because of the expected NASA research interest, the ADS-B model was designed to enable significant levels of message flexibility. The degrees of freedom that the researcher has in the FFSim Phase 1R ADS-B model include:

- Message content
- Number of message types
- Update rate (per message type)
- Nominal range
- Datalink choice (e.g., Ideal, Mode S, or VDL-4)
- Transmission and receipt failures (both scripted and commanded).

The ADS-B message content available allows the researcher to choose any combinations of the 49 data elements shown in Table 3-1. These ADS-B data elements were chosen based on sources from aviation industry [RTCA242, RTCA260], previous NASA research [A99], and emerging ADS-B research discussions [RB00].

Table 3-1 ADS-B Data Elements

ID Data:				
Category	Call Sign	CID	Equipage Level	Beacon Code
Aircraft Type	Aircraft Status	Surveillance Status		
Position Data:				
Latitude	Longitude	Barometric Altitude	Geometric Altitude	Navigational Accuracy Category - Position
Navigational Integrity Category - Position	Geometric Position Valid			
Velocity Data:				
Ground Speed	True Airspeed	Indicated Airspeed	Calibrated Airspeed	North Velocity
East Velocity	Heading	True Ground Track Angle	Magnetic Ground Track Angle	Mach Number
Navigational Accuracy Category - Velocity	Barometric Altitude Rate	Geometric Altitude Rate		
Acceleration Data:				
Turn Indication				
Intent Data:				
Number of Trajectory Change Points (TCP)	TCP Type	TCP Name	TCP Longitude	TCP Latitude
TCP Barometric Altitude	TCP Geometric Altitude	TCP Estimated Time of Arrival	TCP Time to Go	TCP True Airspeed
TCP Indicated Airspeed	TCP Calibrated Airspeed	TCP Ground Speed	TCP Mach	TCP Magnetic Course To
TCP Magnetic Course From	Mode Control Panel (MCP) Selected Heading	MCP Selected Altitude	MCP Selected CAS/Mach	MCP Selected Vertical Rate/Flight Path Angle
Time Data:				
UTC Date	UTC Time of Applicability			

The flexible ADS-B message content allows FFSim researchers the ability to build custom ADS-B messages or mimic existing real-world, standard ADS-B messages such as those defined in Figure 3-2.

ADS-B REPORT ELEMENTS

STANDARD MODE-S ADS-B MESSAGES	"State Vector"														"Mode Status"					"On Condition"		
	ICAO Address	Latitude	Longitude	Altitude (Geodetic)	NUC(p)	North Velocity	East Velocity	Vertical Rate (Geometric)	NUC(r)	Altitude (Barometric)	Barometric Altitude Rate	Air Speed	Ground Speed, Ground Tra	Turn Indication	Call Sign	TCP Latitude	TCP Longitude	TCP Altitude	Aircraft Status (Emergency)	TCP+1 latitude	TCP+1 Longitude	TCP+1 Altitude
Position Squitter	24	17	17	12	5					12												
Velocity Squitter	24			8		11*	11*	10	3		10	11*	22	2								
ID Squitter	24														48							
Intent Type A Squitter	24															14	14	10				
Intent Type B Squitter	24																			14	14	10
Status Squitter	24																		3			

Notes: Entries indicate the number of bits in a message.
The 17 bits of Latitude and Longitude are compressed, and when uncompressed, correspond to a resolution of 24 bits each.
* If velocity over the ground is not available, airspeed and heading are transmitted instead.
Either Barometric or Geometric Altitude is transmitted. The user shall be prompted for which type is to be transmitted.

DEFAULT VALUES

REPORT	DEFAULT UPDATE RATE
Position Squitter	Every 2 Seconds
Velocity Squitter	Every 2 Seconds
ID Squitter	Every 5 Minutes
Intent Type A Squitter	Every 2.5 Minutes
Intent Type B Squitter	Every 2.5 Minutes
Status Squitter	Every 5 Minutes

Figure 3-2 Standard Mode-S ADS-B Messages [LET99]

FFSim researchers can generate multiple ADS-B messages through the construction of different ADS-B message types. This allows one to simulate the simultaneous squittering of different ADS-B reports (e.g., individual “position”, “velocity”, and “ID” reports), each with their own defined update rates.

Researchers can specify a nominal ADS-B range and datalink choice (e.g., Ideal, Mode S, or VDL-4) for each aircraft through the SCC Scenario Setup GUI. ADS-B transmission or receipt failures can be either triggered in real-time through the SCC’s Real-time Event Control GUI or according to a script specified in the SCC’s Scenario Setup GUI. The actual communications modeling (e.g., Mode S line-of-sight constraints) that determines whether aircraft ADS-B messages are being received or not is handled inside the centralized, SCC software.

Additional ADS-B functional details can be found in [SJD01].

3.1.2 Software Description

The ADS-B System has two major functions: 1) generating and transmitting ADS-B messages and 2) receiving transmitted ADS-B messages. The two functions can be broken into an ADS-B Transmitter component and an ADS-B Receiver component. The ADS-B Transmitter component is a separate process while the ADS-B Receiver is a library that is linked in by each process that receives ADS-B messages. Figure 3-3 shows the two components and their relationship to other FFSim components.

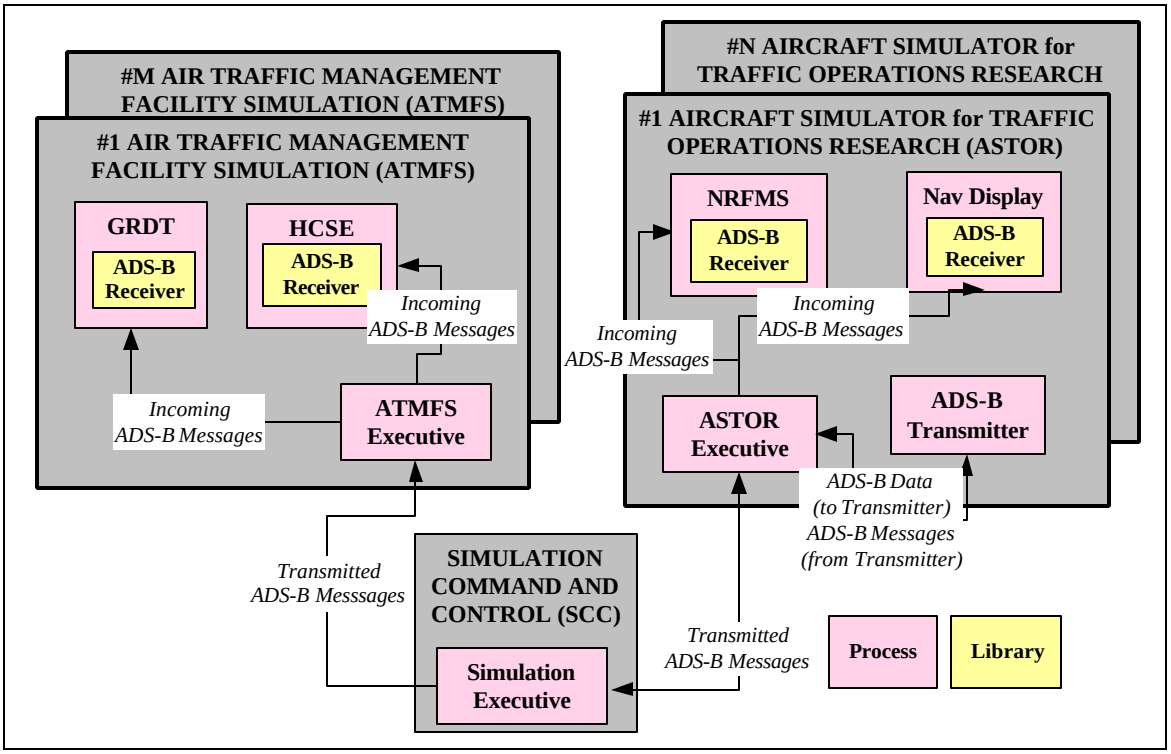


Figure 3-3 ADS-B Receiver and Transmitter Architecture

The ADS-B Receiver is responsible for unpacking ADS-B messages and managing the received data. The ADS-B Receiver package is a statically linked C++ library. Each ASTOR and ATMFS process that needs the receiver's functionality links in the ADS-B Receiver library. The advantage of this approach over a separate process for the ADS-B Receiver is that the inter-process communications of ADS-B messages is significantly reduced. If a separate process were used for the ADS-B Receiver, the ASTOR Executive would have to send incoming ADS-B messages to the ADS-B Receiver which would process them and then send the received ADS-B data back to the ASTOR Executive for consumption by the NRFMS, and Navigation Display processes. With the linked library approach, the ADS-B messages are sent directly from the ASTOR Executive to the NRFMS and FMS displays. The ADS-B Receiver component takes care of receiving and managing the ADS-B messages, however, its functionality is replicated in each process within the ASTOR and ATMFS that needs to receive ADS-B data.

Unlike the ADS-B Receiver component, the ADS-B Transmitter is a separate process. The main reason for this is that the data comprising the ADS-B message comes from several different processes within the ASTOR. The only way to form an ADS-B message is to have a single application that is responsible for collecting all of the necessary data and then determining when to transmit the messages. The ASTOR Executive was a logical candidate for this, however, a goal of the FFSim1R system architecture was to keep the specific domain functionality out of the ASTOR Executive. Another choice could have been to incorporate the ADS-B Transmitter into another existing process, such as the NRFMS. The reason for not choosing this approach was that it would reduce simulation flexibility. If for example, the ADS-B Transmitter was part of the NRFMS that would dictate that each ADS-B equipped aircraft must always be equipped with a NRFMS, which may not always be the case. Having the ADS-B Transmitter as a separate process offers the most flexibility.

The OOD for the ADS-B system was done using the Unified Modeling Language (UML). The entire detailed ADS-B software design is documented in [D01-1]. Figure 3-4 shows the high-level UML packages for the ADS-B system. The Transmitter package contains the classes for the ADS-B Transmitter application and the Receiver package contains the classes for the ADS-B Receiver library. The other packages in Figure 3-4 represent components or libraries that are common to both the Transmitter and Receiver.

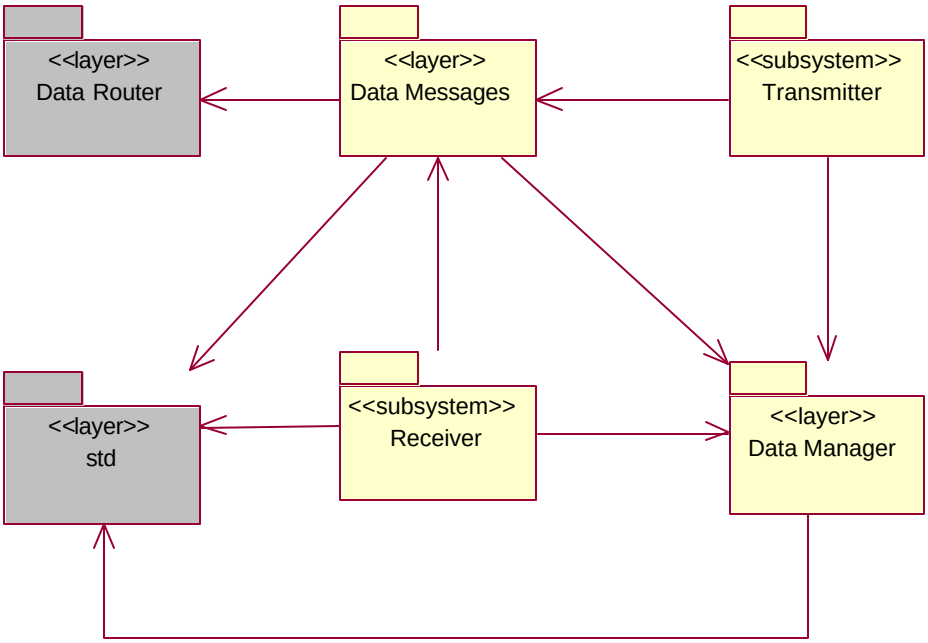


Figure 3-4 ADS-B Software Packages

The Data Manager package is a statically linked C++ library. The purpose of the Data Manager is to manage the individual ADS-B data elements that comprise the ADS-B messages. The ADS-B Receiver and the ADS-B Transmitter both require this functionality. The Data Manager can be thought of as a table in a database. There is a one-to-one relationship between a Data Manager and an individual aircraft. The Data Manager leverages the C++ standard (`std`) library.

The Data Messages package is a statically linked C++ library. The purpose of the Data Messages package is to allow the inter-process communication of the ADS-B data. The Data Messages leverages the existing ASTOR Data Router software package and the C++ `std` library. Both the ADS-B Receiver and ADS-B Transmitter use the ADS-B Data Messages package.

It was desired to make the Data Messages compatible with the existing ASTOR Data Router software. However, the Data Message component design needed to support sending of customized messages which was something that did not exist in any of the other ASTOR messages. The design achieves this by providing a mechanism to generically specify what data comprises the particular ADS-B Message. The Data Messages component then determines how to pack and unpack the data to and from the ADS-B Data Manager. Each supported ADS-B message will have its own ADS-B Message object in each Receiver and Transmitter.

The ADS-B system implementation follows the OOD [D01-1]. The software was developed in C++ adhering to the style guidelines [LAS97].

3.1.3 Further Development Items

The following section describes features that may need to be added or addressed during future integration work.

3.1.3.1 Initialization

Before the ADS-B Transmitter can be used in the simulation it will need to be supplied with an initialization message. The initialization message defines the data elements that compose the desired ADS-B message(s). A temporary message (ADD_ADSB_MESSAGE) is currently in place. Ultimately, the simulation executive will need to generate this message.

3.1.3.2 Data Source Message

The ADS-B Transmitter obtains the data it needs to construct a message by handling incoming ASTOR data through the `SIM_DATA_MSG`, `CDU_DATA_MSG` and `XNET_MSG`. The `XNET_MSG` contains profile array information. Currently this message type is not defined in the general message set and should be when integrated.

3.1.3.3 Transmitter Data Element Integration

Appendix A describes the current status of ADS-B data element software integration with ASTOR. Except for a number of last-minute desired ADS-B element changes, all of the data elements are currently supported in software. However some of the sources for these elements were not available given the current state of the applications responsible for supplying the data. The current state of the intent data elements are the result of a short term integration solution using a combination of ASTOR's `ProfileArrayXnet` data and `CduData` to fill as many elements as possible. See Appendix A for details. The last-minute desired ADS-B element changes included:

- Splitting of "Navigational Uncertainty Category – Position" into 2 new data elements: "Navigational Accuracy Category – Position" and "Navigational Integrity Category – Position"
- Renaming of "Navigational Uncertainty Category – Velocity" into "Navigational Accuracy Category – Velocity"
- Removal of the "TCP Heading" data element
- Replacement of the "TCP True Ground Track Angle" and "TCP Magnetic Ground Track Angle" with "TCP Magnetic Course To" and "TCP Magnetic Course From" data elements
- Renaming of "MCP Commanded Heading" to "MCP Selected Heading"
- Renaming of "MCP Limit Altitude" to "MCP Selected Altitude"
- Addition of the following, two ADS-B intent data elements: "MCP Selected CAS/Mach" and "MCP Selected Vertical Rate/Flight Path Angle"

The ADS-B functional documentation has been to reflect the last-minute ADS-B element changes, but in future ADS-B software development, these changes need to be reflected in the ADS-B software design documentation and software code. An important note regarding the ADS-B data elements is that the standard ADS-B data elements are currently going through a significant revision that is to culminate in a new ADS-B MASPS, DO-242A (see [AD01] for the latest status). Any attempt to reflect the latest ADS-B standards in the ADS-B message set will need to remain responsive to the results from RTCA's SC-186.

3.1.3.4 Receiver Integration

The ADS-B Receiver library needs to be integrated with each airborne and ground application that desires to receive ADS-B messages. The integration involves linking in the ADS-B Receiver library and adding support for the ADS-B message type in the application.

3.1.3.5 Data Link Model

The implemented ADS-B software does not include models for the various communication datalinks, e.g., Mode S. The FFSim 1R architecture calls for these models to be located in the Simulation Command and Control (SCC) component and the models to be developed by another organization. The current ADS-B software is not affected by the omission of these models from the FFSim architecture, nor are any significant changes required to the existing ADS-B system software when the models are completed and integrated.

3.2 Air Traffic Management Facility Simulation Executive

The Air Traffic Management Facility Simulation (ATMFS) portion of the FFSim represents an Air Traffic Management Facility such as an Air Route Traffic Control Center (ARTCC), e.g., Dallas-Fort Worth Center (ZFW). The ATMFS Executive is the central software component of each ATMFS. All inter-process communication with and within the ATMFS is handled by the ATMFS Executive. As part of the Phase 1R effort, the ATMFS Executive was specified, designed and then implemented in C++ on Windows NT and UNIX (Solaris).

The following sections describe the key features of the ATMFS Executive functional design and software.

3.2.1 Functional Description

The ATMFS Executive was originally given seven major functions: Initial Conditions, Startup, Data Router, Data Acquisition, Time Manager, Airport Control Logic, and a Graphical User Interface (GUI) as shown in Figure 3-5. Implementation of the Airport Control Logic was eventually deferred to a future build.

The Initial Conditions function is responsible for communicating the initial conditions to each ATMFS component. The Startup function is responsible for starting each required ATMFS component before the simulation run begins. The Data Router function is responsible for routing all messages sent by ATMFS components to each other or to and from the Simulation Executive. The Data Acquisition function is responsible for acquiring and logging specified ATMFS data. The Time Management function is responsible for synching component's internal clocks to the main FFSim clock. The GUI allows a simulation user (researcher) to interactively control the ATMFS Executive. The Airport Control Logic function allows the researcher to setup and configure airport information, e.g., available runways.

A complete functional description is provided in [DWS01].

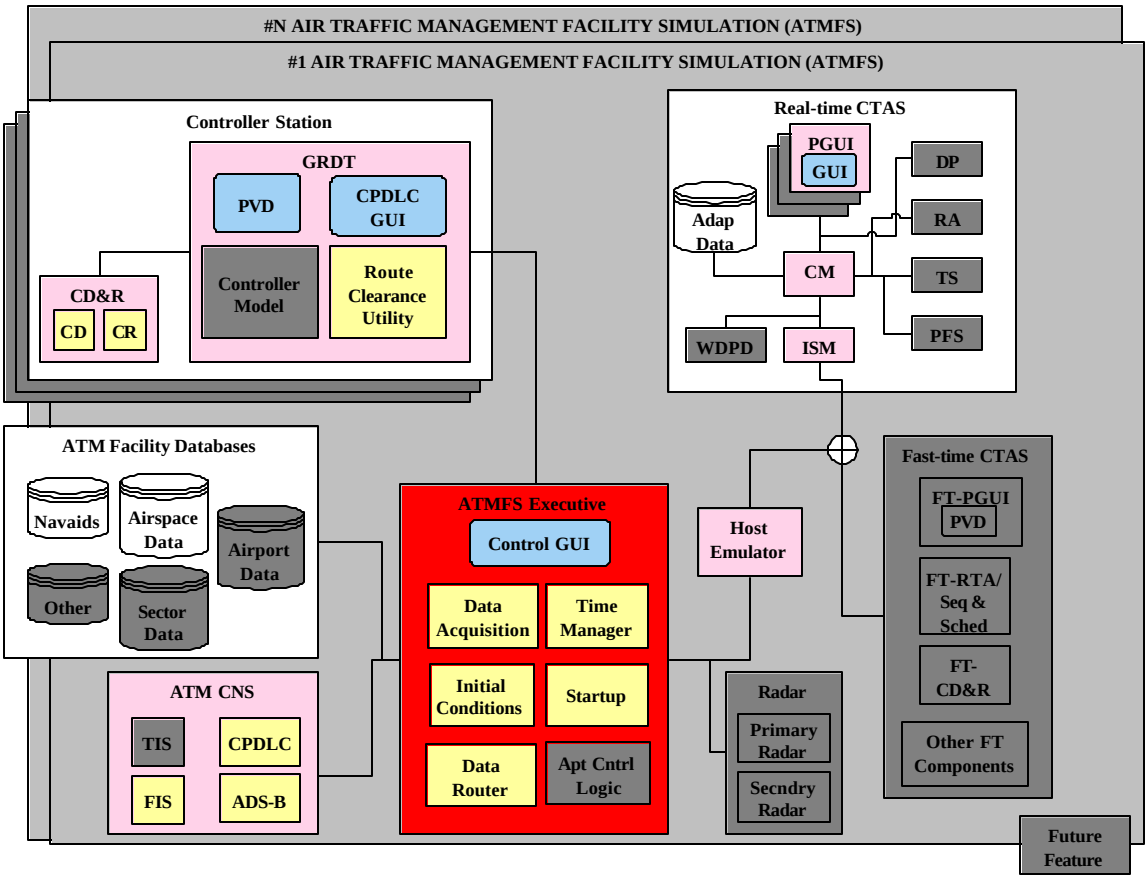


Figure 3-5 ATMFS Functional Diagram

3.2.2 Software Description

The ATMFS Executive software was designed using UML. The design employs several well-known and proven object-oriented design patterns. Figure 3-6 shows the UML class diagram for the ATMFS Executive's state machine. The state machine is responsible for maintaining the current simulation state of the ATMFS Executive and of each ATMFS component. Not only does it keep track of the current state, but it also enforces what logical states can be transitioned to next. Figure 3-6 also shows the use of the Observer design pattern [GHJ95]. The Observer pattern de-couples an object that generates or does something from one or more objects that are interested in when that happens. In the case of the state machine, it is when the current simulation state changes that other objects within the ATMFS Executive are interested.

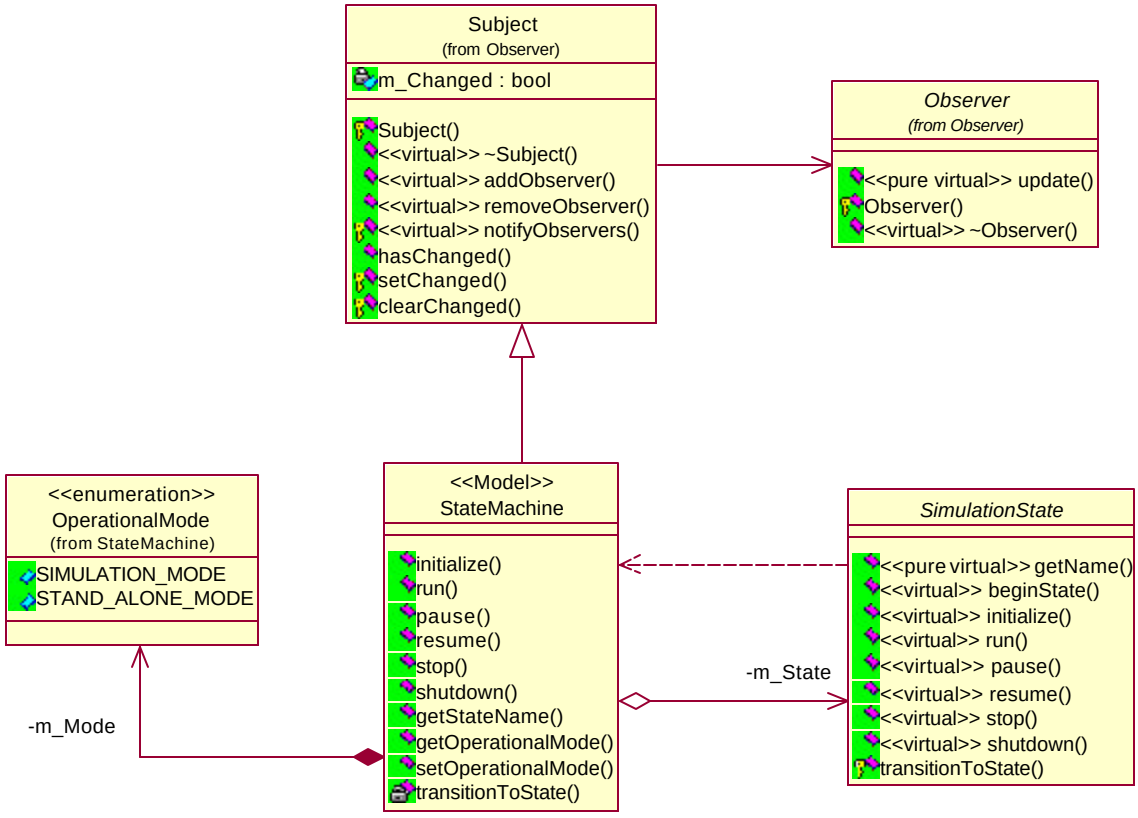


Figure 3-6 UML class diagram of ATMFS Executive State Machine

The main GUI screen for the ATMFS Executive is shown in Figure 3-7. The ATMFS Executive can operate in two modes: simulation and stand-alone. In simulation mode the ATMFS Executive is connected with the rest of the FFSim via the SCC Simulation Executive. This is the normal mode of operation. In the stand-alone mode, the ATMFS Executive is not connected to the Simulation Executive, but it is still connected to the other ATMFS components.

In stand-alone mode, the Air Traffic Management Facility Simulation becomes a stand-alone simulation. While this provides limited research capability, it does permit development and debugging to be done more effectively, especially when other necessary FFSim simulation components are not sufficiently mature. Figure 3-7 shows the ATMFS Executive in stand-alone mode. In this mode, the simulation control buttons (“Initialize”, “Start”, “Pause”, “Stop”) are displayed on the GUI. These buttons allow the ATMFS Executive user to operate the ATMFS Executive as if it were the Simulation Executive. In stand-alone mode the user can also select a time factor (2x, 4x, or Nx) to put the simulation into a fast-time mode. Real-time occurs when a factor of 1x is used. The current simulation time is displayed next to the time factor selection buttons. In simulation mode the Simulation Executive controls the simulation states and the time factor.

The ATMFS Executive main screen displays the current state of the simulation in the Status field. The Status field is supplemented with a red-yellow-green status indicator. When the simulation is stopped the status indicator displays red. When the simulation is initialed or paused the color displayed in yellow. Green signifies that the simulation is running. The ATMFS Executive main screen also has a text window that displays errors and other internally or externally generated messages.

Figure 3-7 ATMFS Executive Main Screen

The ATMFS Executive configuration screen is shown in Figure 3-8. The configuration screen allows Data Acquisition to be turned on/off and configured. Other configuration parameters control the generation of debugging information and initializing the other ATMFS components.

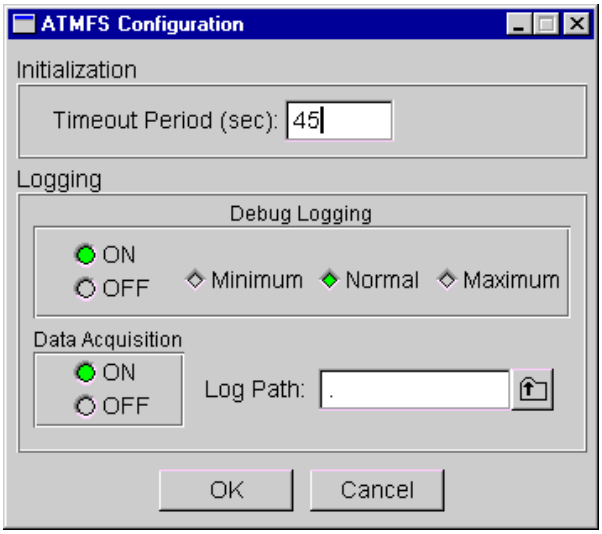


Figure 3-8 ATMFS Executive Configuration Screen

The ATMFS Executive is responsible for starting up the other ATMFS components. The component configuration screen is shown in Figure 3-9. This screen allows the user to add, remove and configure ATMFS components. Each component is provided a name, a host IP address, a path on the host machine where the component resides, and any command line parameters that are needed for the component. The ATMFS Executive component configuration screen also allows the user to select which messages are produced (source) and which messages are consumed (sink) by each component. The ATMFS Executive data router needs the source-sink information in order to correctly route the messages between ATMFS components. The ATMFS Executive was tested using the ADS-B Transmitter, the FIS-B Service, and various test applications.

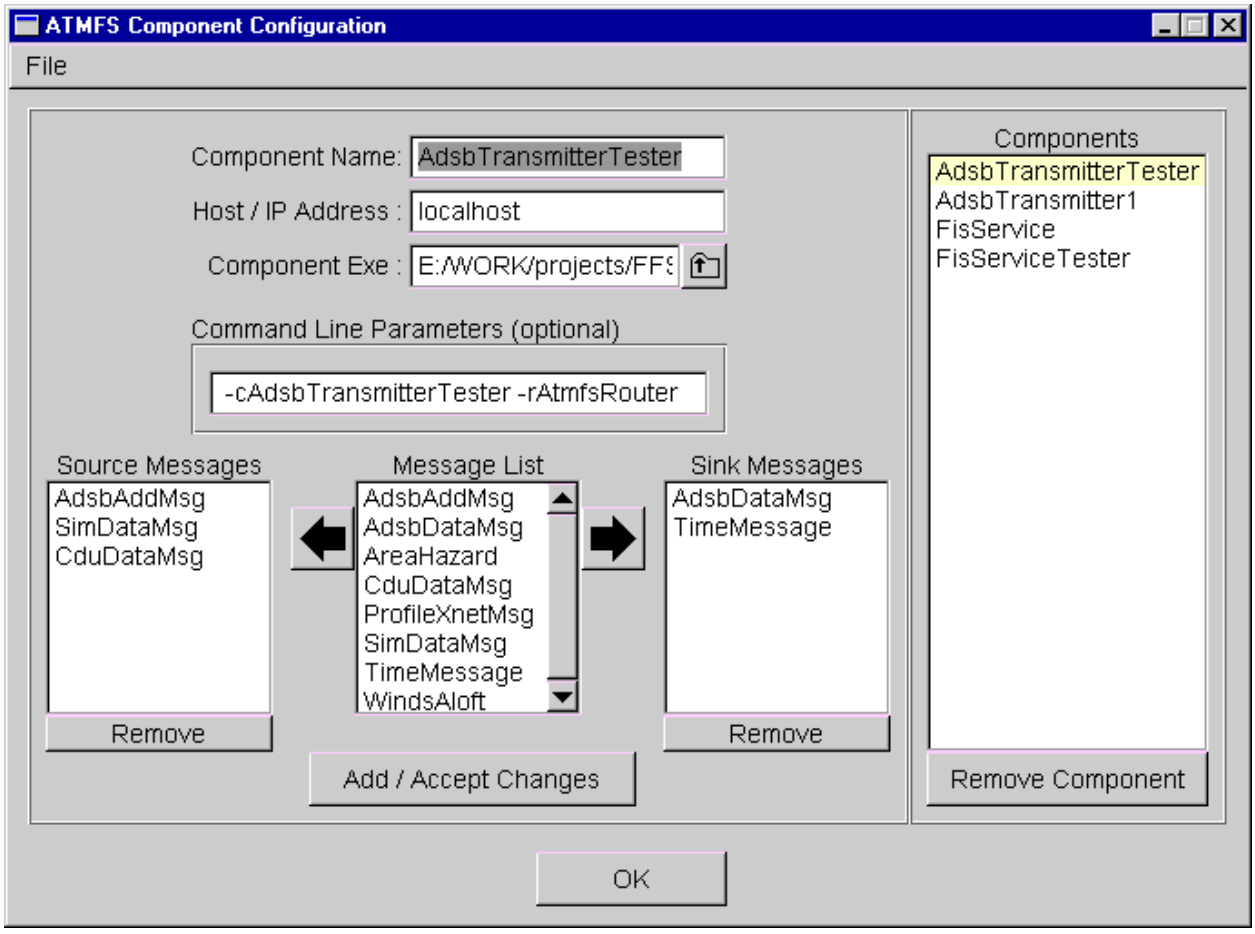


Figure 3-9 ATMFS Executive Component Configuration Screen

The ATMFS Executive software was developed in C++ adhering to the style guidelines [LAS97]. The ATMFS Executive was built and tested on Windows NT/2000 and Solaris.

3.2.3 Further Development Items

3.2.3.1 SCC Integration

The ATMFS Executive must still be integrated with the SCC Simulation Executive. This could not be completed because of the absence of the Simulation Executive. Integration may require that the ATMFS simulation control messages be changed to be consistent with those used by the Simulation Executive and the rest of FFSim.

3.3 Flight Information Services - Broadcast

The Flight Information Services – Broadcast (FIS-B) was designed for the FFSim Phase 1R architecture (shown in Figure 2-3). A detailed functional specification of the Phase 1R FIS-B model was developed and presented to NASA-LaRC [DKJ01]. Following its approval by NASA, a software design was produced and documented in a software design document [D01-2]. Once the software design was reviewed and approved by NASA-LaRC, the FIS-B receiver and transmitter software was implemented in C++ and tested on both Windows NT and UNIX (Solaris) operating systems.

The following sections describe the key features of the FIS-B functional design and software.

3.3.1 Functional Description

The FIS-B component interacts with the other FFSim components as shown in Figure 3-10.

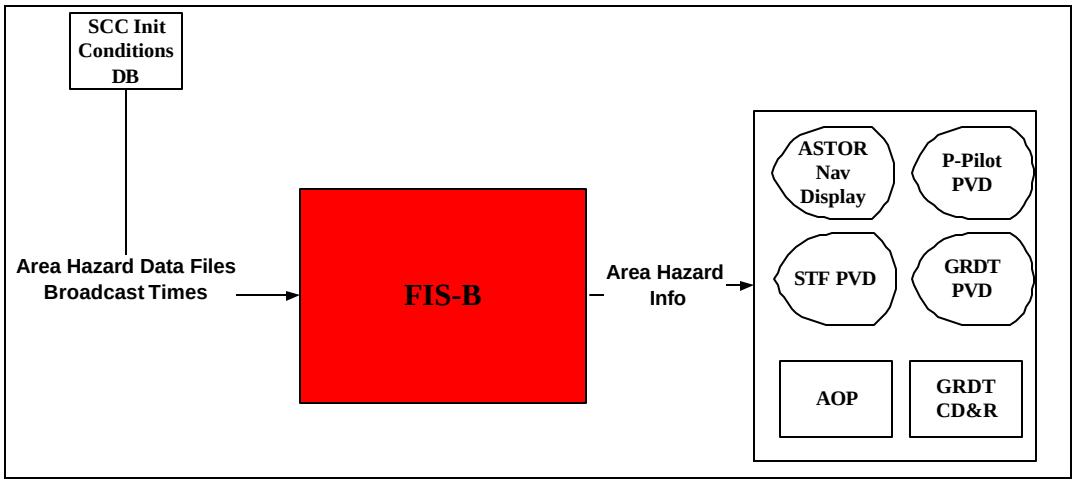


Figure 3-10 FIS-B Inputs and Outputs

The FIS-B component was designed to enable custom specification of multiple, broadcasts of different static or moving area hazards. Originally, SUA and SIGMET messages were separately designed, but based on feedback at the Design Review [MPW00], a single area hazard message format was designed to handle the necessary data elements to represent not only SUA and SIGMETs, but also any other area hazards. Note: discussions in the FAA's Collaborative Decision Making (CDM) program have discussed the potential determination of Flow-Constrained Airspace which could be en route airspace sectors overloaded with traffic that need to be avoided. Such a concept could be adequately modeled in the FIS-B model.

An example representation of the FFSim Area Hazard is shown in Figure 3-11.

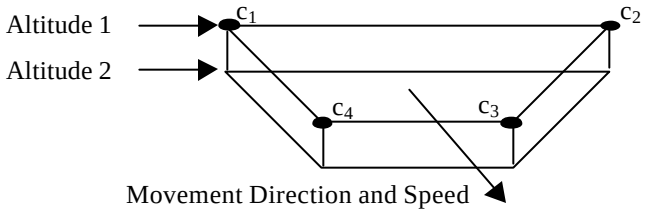


Figure 3-11 FIS-B Area Hazard Illustration

The general Area Hazard allows the researcher to specify the data elements in Table 3-2. These FIS-B data elements were chosen based on sources from aviation industry [RTCA232] [RTCA195] and FIS-B research discussions [MPW00].

Table 3-2 FIS-B Area Hazard message data

Data Item	Definition
Unique Hazard ID	This unique ID is researcher-specifiable and may include hazard type information standards. Examples: SIGZ444 – Icing SIGMET #444 SUAR2513 – Restricted SUA #2513 Real-World Notes: For SIGMETs: • Icing (Z) • Turbulence (T) • Thunderstorms (W) For SUAs: P = Prohibitive R = Restricted A = Alert W = Warning MOA = Military Operations Area For P, R, A, and W types, a number typically identifies the region, for MOAs, a name typically identifies the region. Examples: R-2513 for Hunter-Liggett, CA R-2531 A for Tracy, CA low altitude W-260 for San Francisco, CA MOA-BISHOP for Bishop MOA FOOTHILL 1 for Foothill Northern MOA For SUA
Effective Start Time	UTC Time when the area becomes active
Effective Finish Time	UTC Time when the SUA becomes inactive
Severity	Trace (1); Light (2); Moderate (3); Severe (4)
Altitude 1	Top of hazard layer in hundreds of feet
Altitude 2	Bottom of hazard layer in hundreds of feet
Movement Direction	Direction in tens of degrees
Movement Speed	Speed in knots
Number of Vertices	Number of area hazard vertices
1st Latitude	Latitude coordinate for 1st vertex of watch box polygon in thousandths of a degree; “N” or “S” for Hemisphere
1st Longitude	Longitude coordinate for 1st vertex of watch box polygon in thousandths of a degree; “N” or “S” for Hemisphere
Next Latitude	Latitude coordinate for next vertex of watch box polygon
Next Longitude	Longitude coordinate for next vertex of watch box polygon
etc	“
Last Latitude	Latitude coordinate for last vertex of watch box polygon; user’s software to connect last vertex to 1st vertex (no need to transmit the last with this assumption)
Last Longitude	Longitude coordinate for last vertex of watch box polygon; user’s software to connect last vertex to 1st vertex (no need to transmit the last with this assumption)

Additional details of the FIS-B functionality can be found in [DKJ01].

3.3.2 Software Description

The FIS-B System has two major functions: 1) broadcasting FIS-B messages and 2) receiving FIS-B messages. The two functions can be broken into the FIS-B Service and the FIS-B Receiver components. The FIS-B Service component is a separate process while the FIS-B Receiver is a library that is linked in by each process that receives FIS-B messages. Figure 3-12 shows the two components and their relationship to other FFSim components.

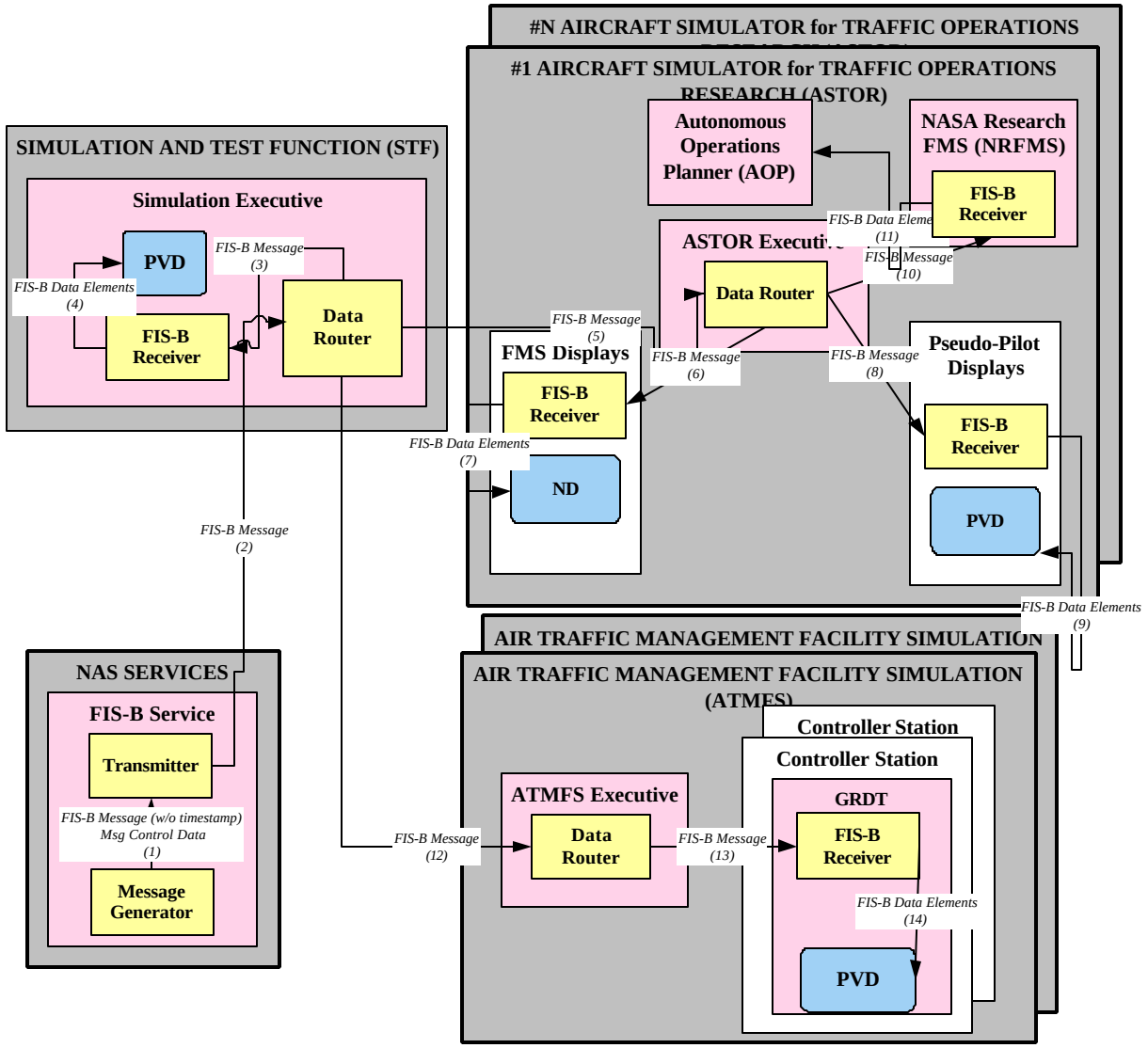


Figure 3-12 FIS-B Receiver and Service Architecture

The FIS-B Service process is responsible for generating and transmitting FIS-B messages with the user-configured content at the user-configured update rate. The SCC Simulation Executive supplies the FIS-B Service with the FIS data needed to construct the FIS-B messages. The FIS-B Service sends the constructed FIS-B messages to the SCC Executive, which forwards them to the ASTOR Executives and the ATMFS Executives. The SCC Executive will contain any future communication models and failure logic.

The FIS-B Receiver library, linked in with each application needing to receive FIS-B messages, handles the unpacking and management of the FIS-B data. The parent application provides the raw data packets that are received via the IPC protocol to the FIS-B library. After unpacking, the FIS-B message data is made available to the parent application by the FIS-B Receiver.

The FIS-B Receiver and FIS-B Service components are dependent on a third component, the FIS-B Messages component. The FIS-B Messages contain the specific knowledge of what data elements comprise the FIS-B message(s). The Service uses the FIS-B Messages to pack and send the FIS-B messages. The Receiver uses the FIS-B Messages to receive and unpack the FIS-B messages.

The OOD for the FIS-B system was done using UML. The entire detailed FIS-B software design is documented in [D01-2]. Figure 3-13 shows the high-level UML packages for the FIS-B system. The Service package contains the classes for the FIS-B Service and Transmitter application and the Receiver package contains the classes for the FIS-B Receiver library. The other packages in Figure 3-13 represent components or libraries that are common to both the Service and Receiver.

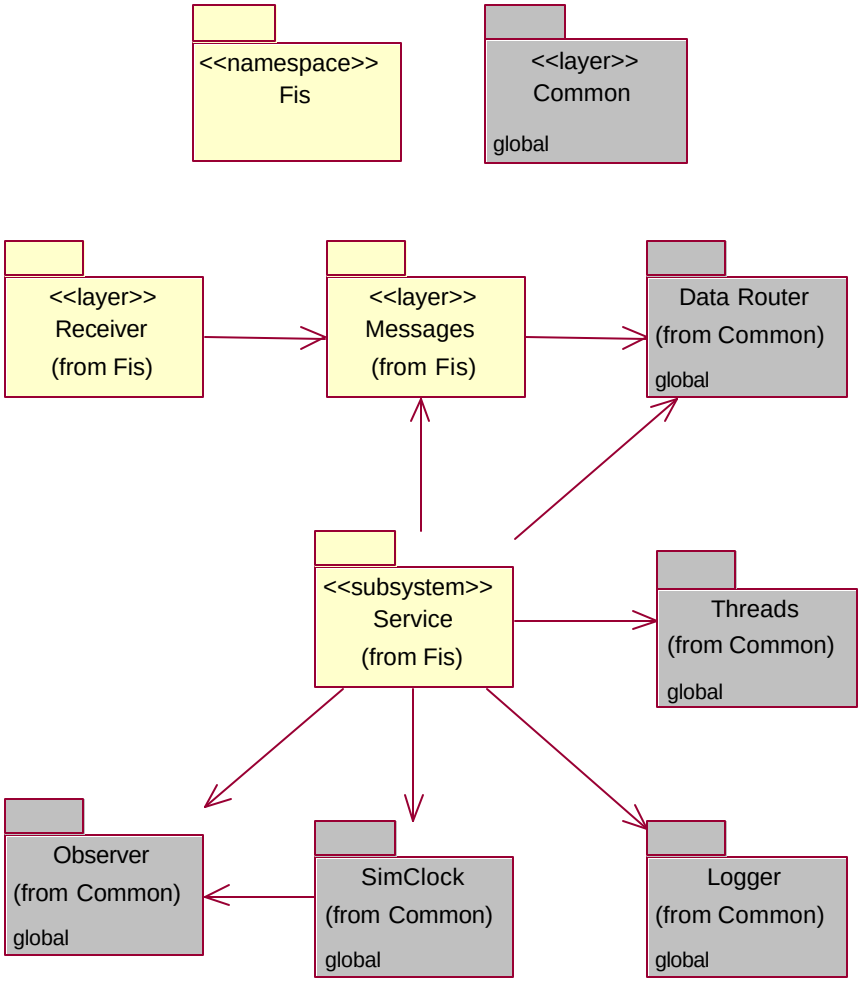


Figure 3-13 FIS-B Packages

The Messages package is a statically linked C++ library. The purpose of the Messages package is to allow the inter-process communication of the FIS-B data. The Messages package leverages the existing ASTOR Data Router software. Both the FIS-B Receiver and FIS-B Service use the Messages package.

All of the FIS-B software resides in the `Fis` C++ namespace. The use of namespaces reduces the chances of name conflicts in future development. The FIS-B system implementation follows the OOD [D01-2]. The software was developed in C++ adhering to the style guidelines [LAS97].

3.3.3 Further Development

The following section describes features that may need to be added or addressed during future work.

3.3.3.1 Integration with ASTOR Components

The FIS-B Receiver package is ready to be integrated with one or more ASTOR applications that need to receive FIS data. The integration involves linking in the FIS-B Receiver library and adding support for the FIS message type to the ASTOR Executive.

3.3.3.2 Integration with ATMFS Components

The FIS-B Receiver package is ready to be integrated with one or more ATMFS applications that need to receive FIS data. The integration involves linking in the FIS-B Receiver library.

3.3.3.3 Implement Winds and Temperatures Aloft

The current implementation of FIS-B can represent area hazards such as dynamic weather regions (SIGMETS) and static or dynamic Special Use Airspace (SUA). Winds and temperature aloft support needs to be added to the FIS-B message, receiver, and service.

3.3.3.4 Data Link Model

The implemented FIS-B system software does not include communication models for air-ground datalinks, e.g., UAT. The FFSim1R architecture calls for these models to be located in the Simulation Command and Control (SCC) component and the models to be developed by another organization. The current FIS-B software is not affected by the omission of these models from the FFSim1R architecture, nor are any significant changes required to the existing FIS-B system software when the models are completed and integrated.

3.4 Controller-Pilot Data Link Communication

The Controller-Pilot Data Link Communication (CPDLC) component was designed to integrate into the FFSim Phase 1R architecture (shown in Figure 2-3). A detailed functional specification of the Phase 1R CPDLC model was developed and presented to NASA-LaRC [SW01]. Following its approval by NASA, a software design was produced and documented in a software design document [DW01]. The CPDLC pilot and controller transceiver software was implemented in C++ and tested on the Windows NT operating system.

The following sections describe the key features of the CPDLC functional design and software.

3.4.1 Functional Description

The CPDLC component’s interaction with the other FFSim components is shown in Figure 3-14.

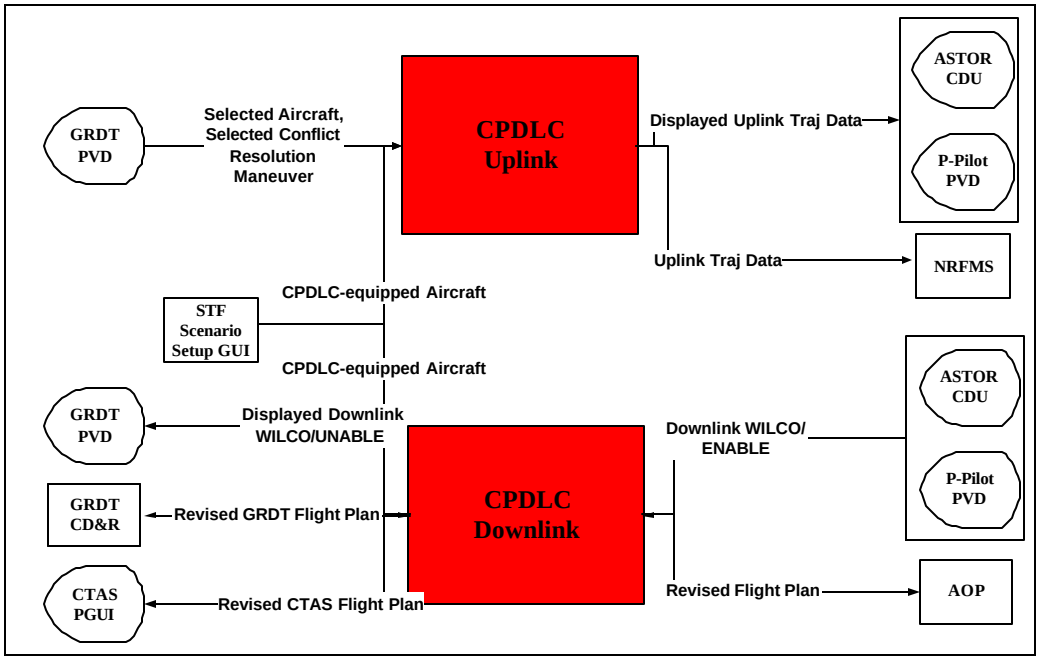


Figure 3-14 CPDLC Model Inputs and Outputs

The CPDLC messages consist of a future, CPDLC Build 2, trajectory clearance uplink message and a pilot response downlink message. The CPDLC message content consists of the data elements shown in Table 3-3. These CPDLC data elements were chosen based on sources from aviation industry [RTCA219], previous NASA research [A99], and feedback from the NASA-Langley Aircraft Systems and Operations research team [MPW00].

Table 3-3 CPDLC Data Elements

Trajectory Clearance Uplink:				
Message Source	Time Stamp	Message ID Number	Message Reference Number	Response Required Flag
Number of Waypoints	Waypoint Name	Waypoint Latitude	Waypoint Longitude	Constraint Altitude
Constraint Speed	Constraint Time			
Pilot Response Downlink				
Message Type	Message Source	Message Destination	Time Stamp	Message ID Number
Message Reference Number	Pilot Response			

The complete CPDLC functional specifications can be found in [SW01].

3.4.2 Software Description

The FFSim CPDLC System has two major functions: 1) broadcasting and receiving Controller CPDLC messages and 2) broadcasting and receiving Pilot CPDLC messages. Each ATMFS and ASTOR can be equipped with a Controller and Pilot transceiver respectively. The CPDLC Controller Transceiver component is responsible for packaging trajectory clearance messages for transmission to a CPDLC equipped ASTOR aircraft. The controller CPDLC messages are sent to the ATMFS Executive, then to the SCC Executive, which forwards them to the ASTOR Executive of the destination aircraft. The CPDLC Pilot Transceiver component is responsible for packaging pilot responses to trajectory clearance messages for transmission back to the source ATMFS controller station. The pilot CPDLC messages are sent to the ASTOR Executive, then to the SCC Executive, which forwards them to the ATMFS Executive of the intended controller.

The OOD for the CPDLC system was done using UML. The entire detailed CPDLC software design is documented in [DW01]. Figure 3-15 shows the high-level UML packages for the CPDLC system. The Controller Transceiver package contains the classes for the Controller Transceiver library and the Pilot Transceiver package contains the classes for the Pilot Transceiver library. The other packages represent components or libraries that are common to both Controller and Pilot Transceivers.

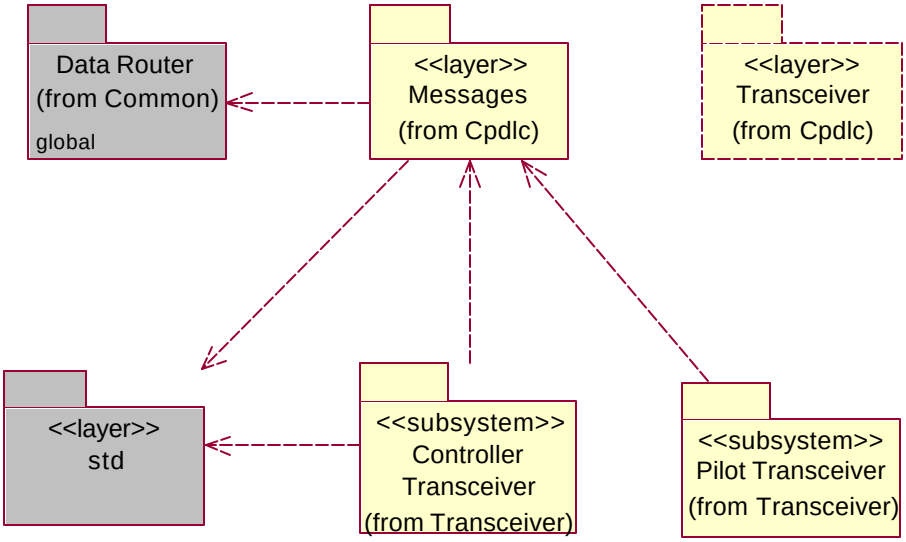


Figure 3-15 CPDLC Software Packages

The Messages package is a statically linked C++ library. The CPDLC Messages package supports the inter-process communication of the CPDLC data. The Messages leverages the existing ASTOR Data Router software. Both the Controller Transceiver and Pilot Transceiver use the Messages package.

The Transceiver package is a statically linked C++ library that is used for sub-classing the Controller Transceiver and Pilot Transceiver classes

The Controller Transceiver package is a statically linked C++ library. The Controller Transceiver package extends the Transceiver package and is responsible for packing Controller messages and receiving Pilot messages. An ATMFS application such as the GRDT would link in the Controller Transceiver library.

The Pilot Transceiver package is a statically linked C++ library. The Pilot Transceiver package extends the Transceiver package and is responsible for receiving Controller messages and packing Pilot response messages. An ASTOR CPDLC application such as the CDU would link in the Pilot Transceiver library.

All of the FIS-B software resides in the `Cpdlc` C++ namespace. The use of namespaces reduces the chances of name conflicts in future development. The CPDLC system implementation follows the OOD [DW01]. The software was developed in C++ adhering to the style guidelines [LAS97].

3.4.3 Further Development

The following section describes features that may need to be added or addressed during future work.

3.4.3.1 Integration with Controller CPDLC Applications

The Controller Transceiver library will need to be integrated with the ATMFS application that is responsible for sending and receiving CPDLC messages. The nominal choice is the GRDT application, but another application could just as well be used.

3.4.3.2 Integration with Pilot CPDLC Application

The Pilot Transceiver library will need to be integrated with the ASTOR application that is responsible for sending and receiving CPDLC messages. The nominal choice is the NRFMS/CDU application, but another application could just as well be used.

3.4.3.3 Data Link Model

The implemented CPDLC software does not include communication models for air-ground datalinks, e.g., VDL-3. The FFSim1R architecture calls for these models to be located in the Simulation Command and Control (SCC) component and the models to be developed by another organization. The current CPDLC software is not affected by the omission of these models from the FFSim1R architecture, nor are any significant changes required to the existing CPDLC system software when the models are completed and integrated.

3.4.3.4 Expand Additional Message

The current FFSim1R CPDLC message set consists of a single controller-generated trajectory clearance message and a pilot response message. Additional messages as defined in [SW01] can easily be implemented in the CPDLC system as needed.

3.5 Host Computer System Emulator

The Host Computer System (HCS) performs flight data processing, radar data processing and message processing at each ARTCC. The Host Computer System acts as the current hub of ARTCC-based data processing. The FFSim Phase 1R architecture contains a Host Computer System Emulator (HCSE). In FY00 functional specifications were generated for the HCSE. No software was designed or implemented. The CTAS Communications Manager (CM) and Input Source Manager (ISM) were investigated in order to determine how best to integrate the HCSE with CTAS. More details can be found in [SCD01]. The NASA-ARC Airborne Operations Lab (AOL) was also consulted about CTAS integration. The AOL has integrated CTAS with the Aeronautical Datalink and Radar Simulator (ADRS) application. There are portions of the ADRS that should be leveraged in implementing the HCSE.

3.5.1 Functional Description

The HCSE emulates the Host Computer System as part of the FFSim Air Traffic Management Facility Simulation (ATMFS); however, the role of the HCSE in the FFSim is not entirely analogous to the real world HCS. The HCSE also serves as a communication interface between the ATMFS Executive and CTAS as shown in Figure 3-16.

An important role of the HCSE is sending and receiving CTAS input and output messages. The HCSE constructs messages for transmission to CTAS by assembling, interpreting and reformatting data originally generated by other FFSim components. The HCSE receives flight data for flights generated by the ASTOR component and track data derived from the Radar process or the ADS-B components of the CNS. The HCSE also receives simulation control commands from the ATMFS Executive.

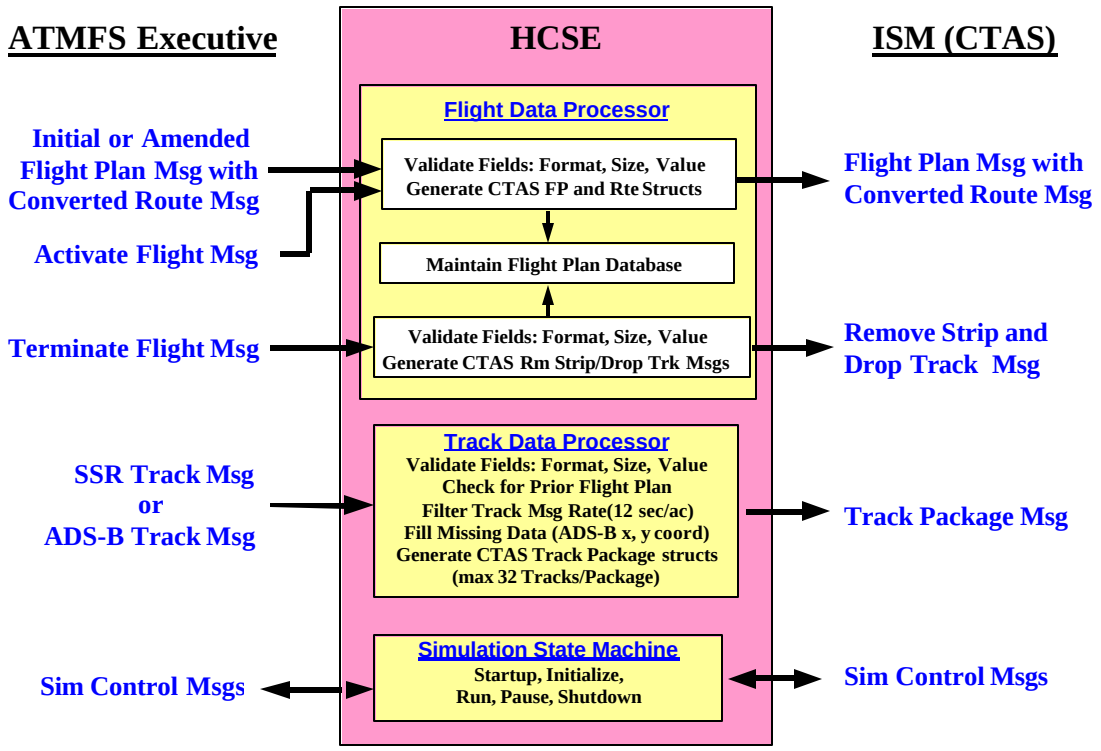


Figure 3-16 HCSE Functions and Data Processing

The Flight Data Processor processes flight plan, converted route and flight activation and termination data sent from the ATMFS Executive, and compiles, enhances if necessary, and transmits these data to the CTAS ISM. The Flight Data Processor performs the calculations necessary to maintain the integrity of the flight data provided to CTAS by implementing the following capabilities:

- Flight plan and converted route processing
- Flight plan database maintenance
- Flight termination

The ATMFS Executive forwards aircraft Track Data messages to the HCSE. Valid and qualified track data are then forwarded to the ISM for use by CTAS. The Track Data Processor validates the track messages, checks for an existing flight plan, filters the track data and then sends the track to CTAS.

The Simulation State Machine controls the state of the HCSE. The State Machine receives simulation control messages from the ATMFS Executive. The State Machine must also control the state of CTAS through existing or new control messages.

For further details on HCSE functions refer to [SCD01].

3.5.2 Further Development

The following section describes features that need to be addressed during future work.

3.5.2.1 Software Design

An object-oriented design needs to be generated for the HCSE if the HCSE is to be implemented from scratch in C++.

3.5.2.2 Implementation

The HCSE will need to be implemented in C++ on Windows NT and UNIX. The NASA-ARC AOL ADRS application should be leveraged where possible.

3.6 Mode S Datalink Communication

The Mode S link operates in two basic ways. The first way is that a request for data is sent to any or all aircraft on one frequency (1030 MHz) and the aircraft within range respond with their Mode S address and data on another frequency (1090 MHz). The data that can be sent by the Mode S transponder are obtained from 256 (56-bit) data registers, which are filled by the aircraft FMS. This mode is the standard Secondary Surveillance Radar (SSR) air-to-ground mode, or Automatic Dependent Surveillance – Address (ADS-A or just ADS) mode. It can also be used between aircraft. The SSRs have a range of about 50 nm in the Terminal Radar Approach Control (TRACON) and 200 nm in the Air Route Traffic Control Center (ARTCC) airspace. The general system description is illustrated in Figure 3-17.

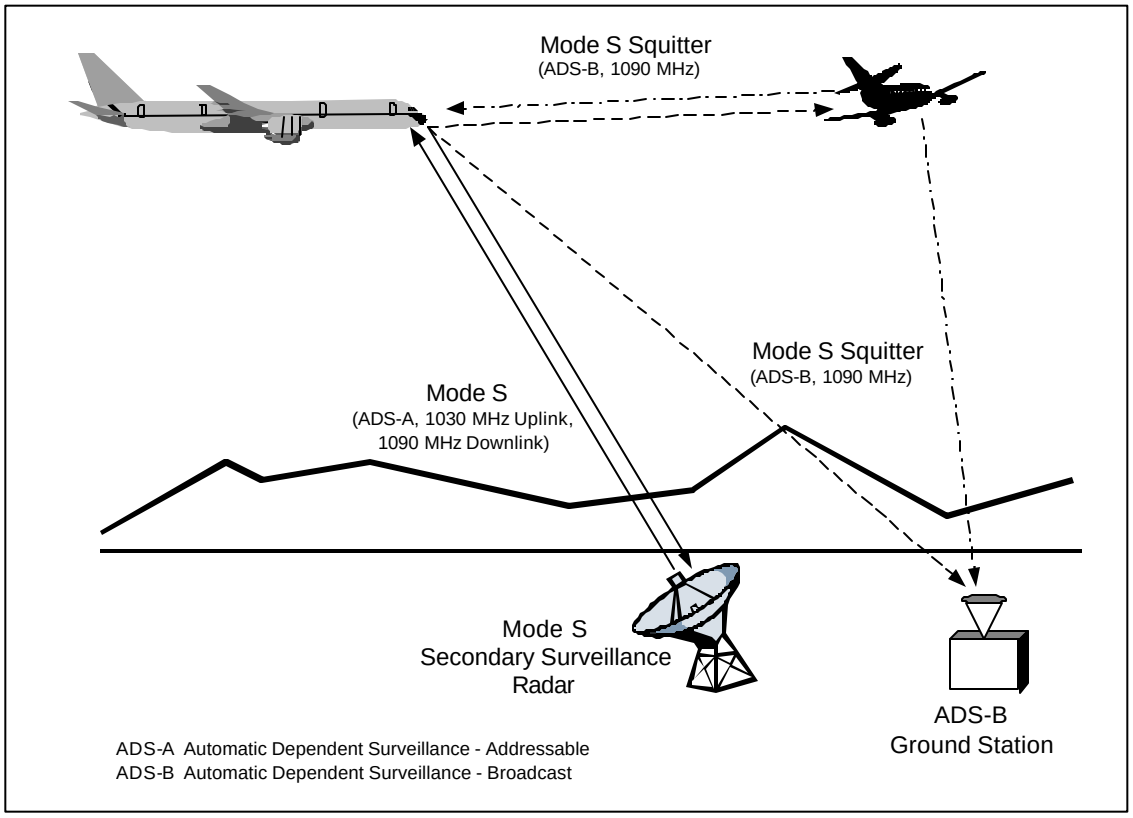


Figure 3-17 Air-Ground Mode S and Mode S Squitter Operations

The aircraft's Mode S transponder automatically and periodically broadcasts, or 'squitters', aircraft data on one frequency (1090 MHz). This mode of operation is called Automatic Dependent Surveillance – Broadcast (ADS-B). This is the primary air-to-air mode between aircraft.

In addition to the Mode S transmissions, the same broadcast frequency is used by the Air Traffic Control Radar Beacon System (ATCRBS) Mode A and C (20 μsec) transmissions as well as the Traffic Alert and Collision Avoidance System (TCAS). While the ATCRBS and Mode S use different transponders (usually an aircraft will have one or the other), the Mode S transponder replies to all ATCRBS interrogations. With TCAS, an aircraft broadcasts its ID at a high update rate to all aircraft within a short range (~14 nm). Neighboring aircraft that receive this transmission then discretely interrogate the transmitting aircraft to determine whether there will be a conflict.

Since all these broadcasts are occurring on the same frequency (1090 MHz), the potential for message collisions is likely. A message collision occurs if the broadcast time interval from two separate messages on the same broadcast frequency overlap partially or completely. To minimize these message collisions, the Mode S ADS-B update rates for repeated data broadcasts are therefore randomly offset about a nominal broadcast interval. This minimizes the potential for message collisions in case another aircraft repeatedly uses the same broadcast time slots. In addition, the Mode S error correction logic has been designed such that it is able to correct for one ATCRBS Mode A or C message collision.

The standard message formats that are used by Mode S consist of a short (56 bit, 64 μsec) message or a long (112 bit, 120 μsec) message. The long message generally contains 56 bits for data, in addition to the standard message overhead (8 bits for control, 24 bits for Mode S address, and 24 bits for parity). However, a special form of the long (112 bit, 120 μsec) message includes 80 bits of data, by reducing some of the message overhead. This message, which does not contain surveillance data, is called an Extended Long Message (ELM).

Since the standard long message is still not long enough to allow all of the ADS-B data to be broadcast at the same time, multiple broadcasts are required to broadcast the full ADS-B message.

3.6.1 Functional Description

The Mode S communications link module resides in the SCC as shown in Figure 3-18. The ASTOR generates the ADS-B message and the ASTOR Executive sends it to the SCC Executive. The message is passed to the Mode S module in the SCC. The Mode S module determines whether that message will be received, error free or not at all, by the remaining ASTORs (2 – N) as well as the ATMFS.

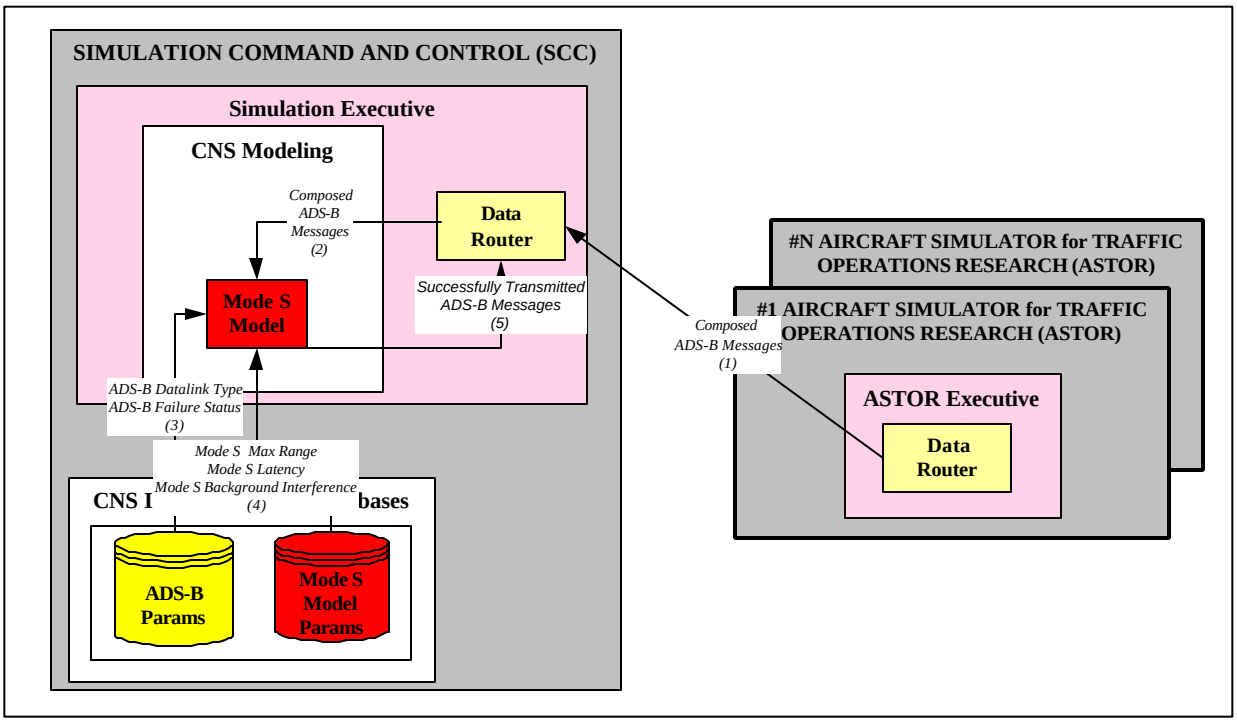


Figure 3-18 Relationship between ADS-B message transmission and Mode S functions

The Mode S subsystem is initialized from the CNS Initial Conditions Database. The Mode S Model calculations determine whether ADS-B message recipient is within max broadcast range and within line-of-sight (LOS) of the broadcast aircraft, compute probability that the ADS-B message will reach recipient error-free, and use probability to determine which ADS-B messages will reach recipients error-free.

For further details on Mode S functionality refer to [MS01-1].

3.6.2 Further Development

The following section describes features that need to be addressed during future work.

3.6.2.1 Software Design

A design needs to be generated for the Mode S module. Ideally the design would be object-oriented. However, due to implementation constraints, i.e., integration into an existing non-C++ application, a procedural design might be necessary.

3.6.2.2 Implementation

The Mode S module will need to be implemented on Windows NT and UNIX. If the SCC Executive used is NLR's TEM, which is written in FORTRAN, additional implementation issues arise. Nevertheless the preferred implementation language would be C++ or possibly C. The Mode S module should be implemented as a library. This can then be linked in the FORTRAN code if necessary. The advantage of using C++ is that it will be easier to reuse the module in new or different versions of the SCC, e.g., the ADRS.

3.7 VDL-4 Datalink Communication

The VHF Data Link (VDL) Mode 4 (VDL-4) data link is used primarily to broadcast aircraft position, velocity, and aircraft intent information to other aircraft or a ground station, as illustrated in Figure 3-19. This concept is called Automatic Dependent Surveillance – Broadcast (ADS-B). It also allows point-to-point messages to be transmitted between aircraft or between an aircraft and a ground station. These latter communications may be 2-way communications, consisting of a request for data by an aircraft or ground station followed by a transmission of this requested data.

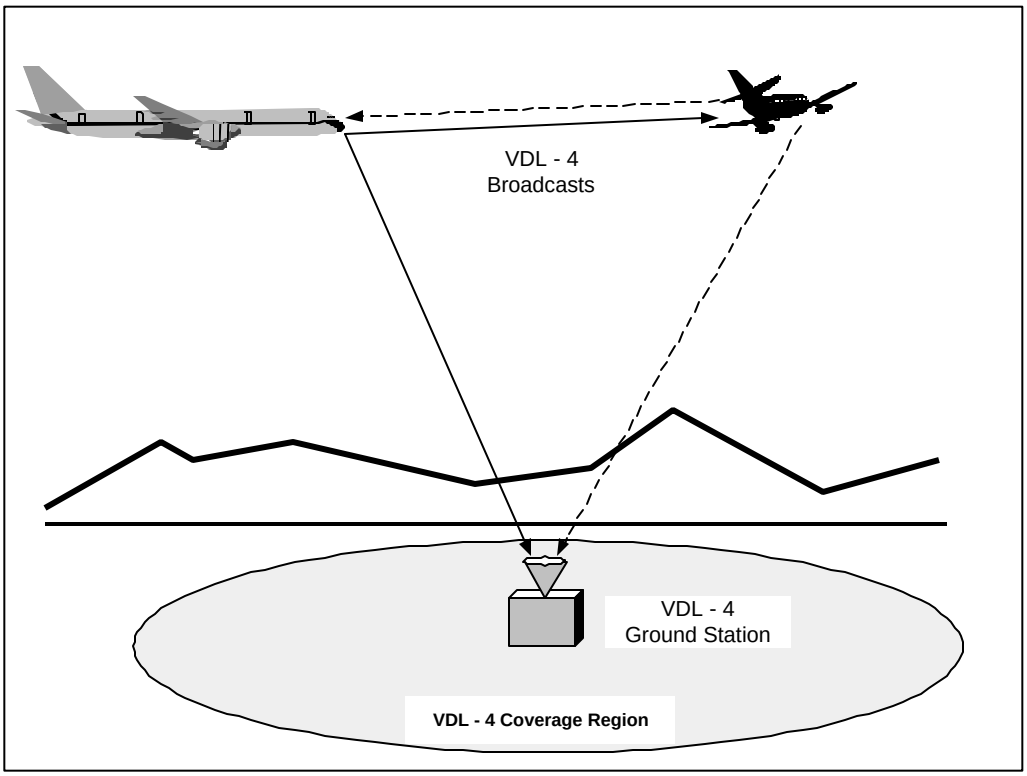


Figure 3-19 VDL-4 Broadcast Operations

While the concept of VDL-4 has been discussed since the 1970's, the formal definition of the architecture has not yet been fully specified either by an RTCA committee or the FAA. However, the International Civil Aviation Organization (ICAO) has draft standards that are currently under review. In addition, VDL-4 is currently in operational use by North European ADS-B Network (NEAN). It has also been selected by Russia to satisfy its future aviation surveillance needs. The following description will focus on only the key features of the VDL-4 system and the ADS-B message broadcast capabilities of this data link.

The basic VDL-4 link consists of N_{GSC} General Signaling Channels (GSC) that consist of a super frame that is one-minute long. Each super frame has 4500 message time slots (for a broadcast or point-point message) with the Gaussian Filtered Frequency Shift Keying (GFSK) modulation scheme. Each message slot contains 256 bits, of which 192 bits are available for data.

Aircraft that want to broadcast use a self-organizing time division multiple access (STDMA) message reservation protocol. This means that each aircraft selects its own broadcast time slot using a specified message slot reservation procedure. Hence, a ground station is not required to handle the message slot assignments.

As an aircraft enters a coverage region, it listens for one minute (one super frame) to determine the existing broadcast reservations on the GSC to which its transmitter is tuned. The start point of listening is any 13.33 ms message slot. The aircraft identifies all the occupied slots in addition for how many super frames these slots have been reserved (included in VDL-4 broadcast message). In addition, it computes the distance to all aircraft that have broadcast on the GSC, to determine which aircraft are more than 250 nm away.

From this survey the aircraft now has a choice to reserve any of the unused channels or any of the channels used by aircraft more than 250 nm away. It select a specific set of m time slots in a super frame on its broadcast GSC. Next, it selects M super frames (3-8) that it will use to broadcast in these specific m time slots. As it reaches the end of its M super frames, it listens (when not broadcasting) and identifies another set of m time slots which are offset from its current slots that are available and to which it can switch. Finally, it selects the message slot offset, Dt, for the m time slots that it will use for the next M super frames. The process can be repeated indefinitely through the coverage region of a particular ground station as well as between coverage regions.

With a capacity of 4500 messages per minute, the message slot rate is 75 slots/sec for each GSC. Since each GSC has a bandwidth of 25 KHz in the frequency band between 108 – 136.975 MHz, there are a maximum number of 1159 possible GSC's. However, nominally only 2 GSCs will be used, while 4 may be used in congested areas to provide sufficient message traffic capacity. Each aircraft and ground transceiver must be able to transmit on at least one GSC and be able to receive on all GSCs. Hence, the standard aircraft transceiver will probably consist of one transmitter with 2-4 receiver channels.

The VDL-4 data link operates in the megahertz frequency band (108 – 136.975 MHz). It is a line-of-sight communications link that is blocked by the earth surface and any other physical object. The maximum line-of-sight range, for two aircraft flying at an altitude of 30 kft, is 370 nm, while the aircraft can be seen from a ground site at only half of this range. In addition, a ground station may have an additional minimum elevation angle constraint.

The maximum distance that this communications link can be used, however, will be determined by transmitter and receiver power levels as well as the signal power losses that occur when transmitting a signal through the atmosphere.

3.7.1 Functional Description

The VDL-4 communications link module resides in the SCC as shown in Figure 3-20. The ASTOR generates the ADS-B message and the ASTOR Executive sends it to the SCC Executive. The message is passed to the VDL-4 module in the SCC. The VDL-4 module determines whether that message will be received, error free or not at all, by the remaining ASTORs (2 – N) as well as the ATMFS.

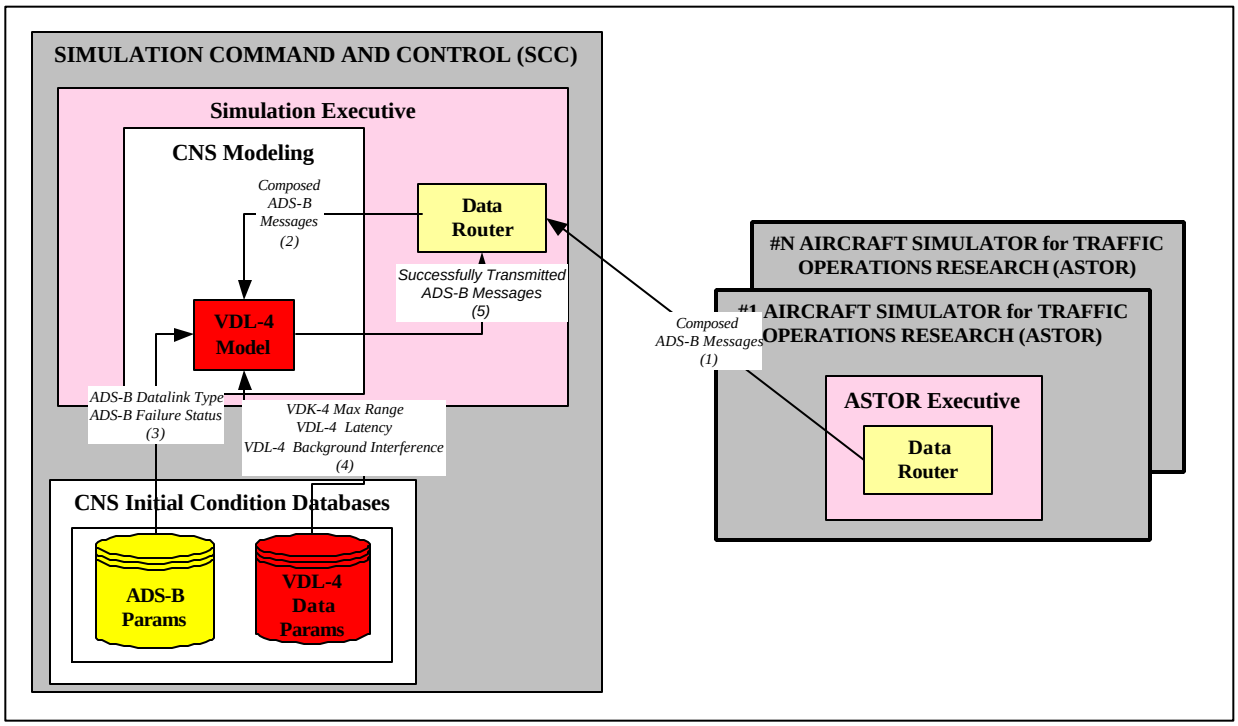


Figure 3-20 Relationship between ADS-B message transmission and VDL-4 functions

The VDL-4 subsystem is initialized from the CNS Initial Conditions Database. The VDL-4 Model calculations determine whether ADS-B message recipient is within max broadcast range and within line-of-sight of the broadcast aircraft, compute probability that the ADS-B message will reach recipient error-free, and use probability to determine which ADS-B messages will reach recipients error-free.

For further details on VDL-4 functionality refer to [MS01-2].

3.7.2 Further Development

The following section describes features that need to be addressed during future work.

3.7.2.1 Software Design

A design needs to be generated for the VDL-4 module. Ideally the design would be object-oriented; however, due to implementation constraints a procedural design might be necessary.

3.7.2.2 Implementation

The VDL-4 module will need to be implemented on Windows NT and UNIX. If the SCC Executive used is NLR's TEM, which is written in FORTRAN, additional implementation issues arise. Nevertheless the preferred implementation language would be C++ or possibly C. The VDL-4 module should be implemented as a library. This can then be linked in the FORTRAN code if necessary. The advantage of using C++ is that it will be easier to reuse the module in new or different versions of the SCC, e.g., the ADRS.

3.8 Secondary Surveillance Radar

Primary Surveillance Radar (PSR) and Secondary Surveillance Radar (SSRs) are used in the Terminal Radar Control (TRACONs) near airports and in the en route airspace of the Air Route Traffic Control Centers (ARTCCs). The key difference between the primary and secondary radars are that the former uses skin tracking of the aircraft to determine the aircraft position while the latter uses an Air Traffic Control Beacon Interrogator (ATCBI) to identify the call sign of the aircraft, and determine its position. The focus of the FY00 FFSim radar simulation capability was on the SSRs. Figure 3-21 illustrates the operation of the SSRs.

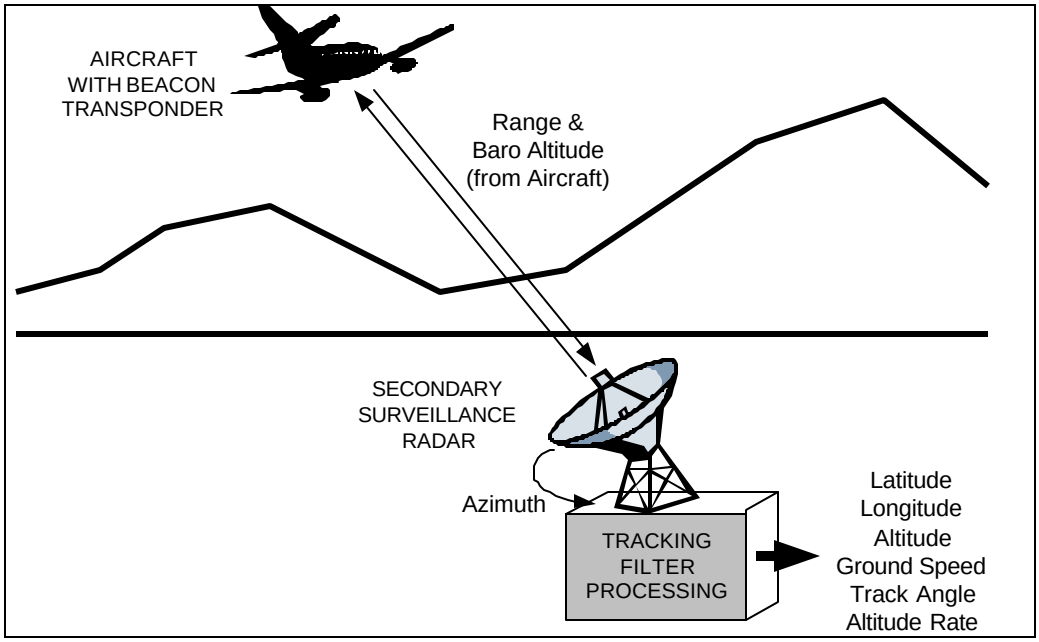


Figure 3-21 Secondary Surveillance Radar Operation

The SSR consists of a rotating antenna dish whose azimuthal direction provides one of the measurements. The TRACON SSRs complete a single rotation in about 4.5 seconds while the ARTCC SSRs require 10-12 seconds. The range to the aircraft ATCBI is obtained by sending a radar pulse to the aircraft. This is amplified and returned together with the aircraft ID (Mode A, C, and S) and the barometric altitude (Mode C and S) from the aircraft. By multiplying the total time of the roundtrip signal travel by the speed of light and taking into account the two-way path as well as any delays in the ATCBI, the range is determined. The maximum range of the TRACON SSRs is about 55 nautical miles while the ARTCC SSRs have a maximum range of about 200 nautical miles.

These measurements are passed to the Host Computer System (HCS) in the ARTCC and to the Automated Radar Terminal System (ARTS) in the TRACON. The HCS and ARTS, respectively compute the latitude, longitude, geometric altitude, ground speed, track angle, and altitude rate using an alpha-beta tracking filter together with coordinate conversions.

In FY00 functional specifications were generated for the SSR. No software was designed or implemented.

3.8.1 Functional Description

Figure 3-22 shows a more detailed view of the ATMFS and the Radar subsystem (consisting of primary and secondary radar functions).

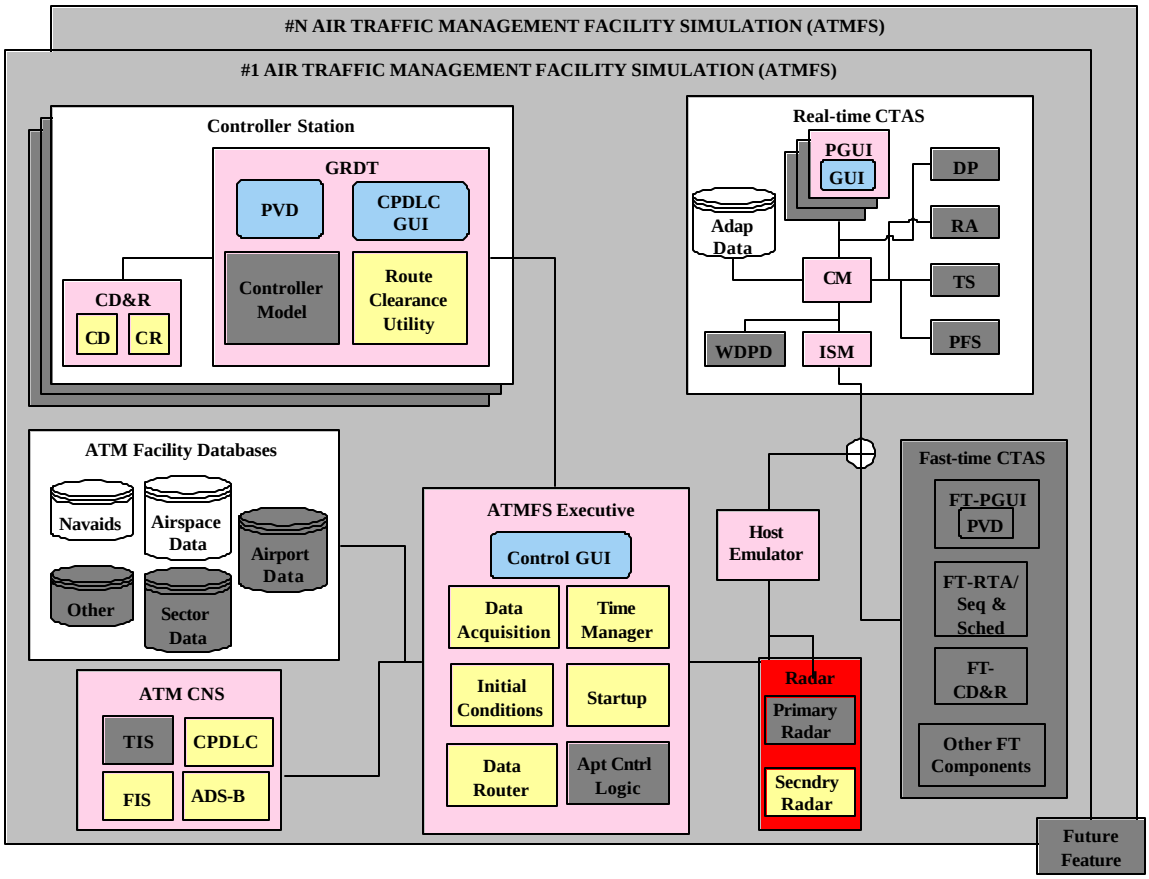


Figure 3-22 A detailed view of the Radar component and its relationship to the ATMFS

The Radar system is composed of subsystems that help it perform its function; that is, to provide radar data to CTAS and GRDT. The functions are the same for both the Primary and Secondary Surveillance Radar systems except for the filtering algorithm. The functional subsystems include Initializer, Data Router, Data Manager, Time Manager, and Tracking Filter.

The Initializer is responsible for initializing all Radar subsystems. The Data Router is responsible for acquiring truth data and initial conditions from ATMFS Executive, for logging specified data, and for routing perturbed positions to clients. The Data Manager is responsible for data conversion and for holding data until time to send it out. The Time Manager is responsible for synching internal clocks to the main FFSim simulation clock. The Tracking Filter is responsible for perturbing truth data with an alpha-beta tracking filter.

For further details on SSR functionality refer to [DMS01].

3.8.2 Further Development

The following section describes features that need to be addressed during future work.

3.8.2.1 Software Design

An object-oriented design needs to be generated for the SSR.

3.8.2.2 Implementation

The SSR will need to be implemented in C++ on Windows NT and UNIX. The NASA-ARC AOL ADRS application should be leveraged where possible.

3.9 Autonomous Operations Planner Recommendations

During the RTO-21 effort, Seagull staff conducted a review of the ongoing AOP functional specs and formulated a recommended high-level AOP-FFSim software architecture. The following sections describe the work performed.

3.9.1 Functional Review

In [C00], NASA identified preliminary AOP functions and functional interfaces with a Research FMS. Revisions were performed to this reference to refine the document with functional and information requirements for:

- State-based Intent Inferencing (Ownship)
- Traffic Information Ambiguity Resolution
- Traffic Data Confidence Assessment
- Traffic Data Fusion
- Traffic Single-State Flight plan/Trajectory Generation
- Traffic aircraft trajectory estimation
- State based Intent Inferencing (Traffic Aircraft)
- Area Hazard Information Ambiguity Resolution
- Area Hazard Confidence Assessment
- Area Hazard Data Fusion

Inputs and outputs to functions were specified or refined, and detailed “shall” statements were incorporated into the functional specifications to capture the requirements for each functionality. To support a three-tiered build schedule, AOP functional and information requirements were identified for Build 1 or noted as grayed-out for Build 2 and 3 items.

To insure continuity between past FFSim work and future Builds, several FFSim related documents were used as the basis for the functional requirements. In particular, the FFSim ADS-B [SJD01], CPDLC [SW01], and FIS-B [DKJ01] specifications were used to identify information requirements. The RTO-30 “Intent Inference, Confidence Assessment, and Hazard Prioritization Status Report” [KMS00] provided additional functional and information requirements.

The final, reviewed functional specifications were provided to Ms. Sheila Conway and are not provided as part of this final report.

3.9.2 Software Architecture Recommendations

According to [C00], "As a part of the Advanced Air Transportation Technologies (AATT) Project of the Aviation System Capacity Program, NASA is performing research on distributed air/ground traffic management (DAG TM)[2]. Part of this research involves development of an Autonomous Operational Planner (AOP) and operational procedures for its use. NASA has identified preliminary AOP functions and functional interfaces with a Research FMS, also under development by NASA. This preliminary design is undergoing review and further development."

Current, accurate data are extremely important to the AOP functions. AOP will use dynamic and static data and must have information regarding the accuracy of the data. Common data that are used by different functions must be exactly the same (within minimal tolerances) and be acquired at the same time. For example, if the conflict detection function is using an ADS-B message from a nearby aircraft when an updated message arrives and then the conflict resolution function uses the new ADS-B message, errors in the resolution may occur.

An AOP software architecture was formulated that contains a Communications Manager, Data Manager and Data Bus to accurately handle all data for the AOP functions.

3.9.2.1 Data

3.9.2.1.1 Static Data

Static data are data that will be stored in databases. (See Figure 3-23 for their position in the AOP Software Architecture.) Generally, these data will not be updated during a simulation run. Examples of static data are:

- Airport data
- Terrain data
- SIDs and STARs
- Navaids

3.9.2.1.2 Initial Conditions

Initial condition data are used to initialize functions and parameters when AOP starts up. Some data may come from a Crew Input Interface and some may come from scenario files (shown as IC Data in Figure 3-23). Some data may be changed later through user input. Examples of initial conditions are:

- Separation standards
- Protected Airspace Zone
- Display options
- Look-head time
- Mode of resolution evocation
- Resolution advisories restrictions
- Start time

3.9.2.1.3 Dynamic Data

These data are generated during a simulation run. The data may come from various components of the simulation at various rates. They may be asynchronous or synchronous events. Examples of dynamic data are:

- ADS-B data
- CPDLC data
- TIS data
- FIS data
- GPS data
- CDTI user inputs (obtained through Crew Input Interface)
- FMS user inputs (obtained through the FMS)

3.9.2.2 Architecture

Figure 3-23 graphically displays the organization of the proposed architecture. This section describes the data handling layers in more detail.

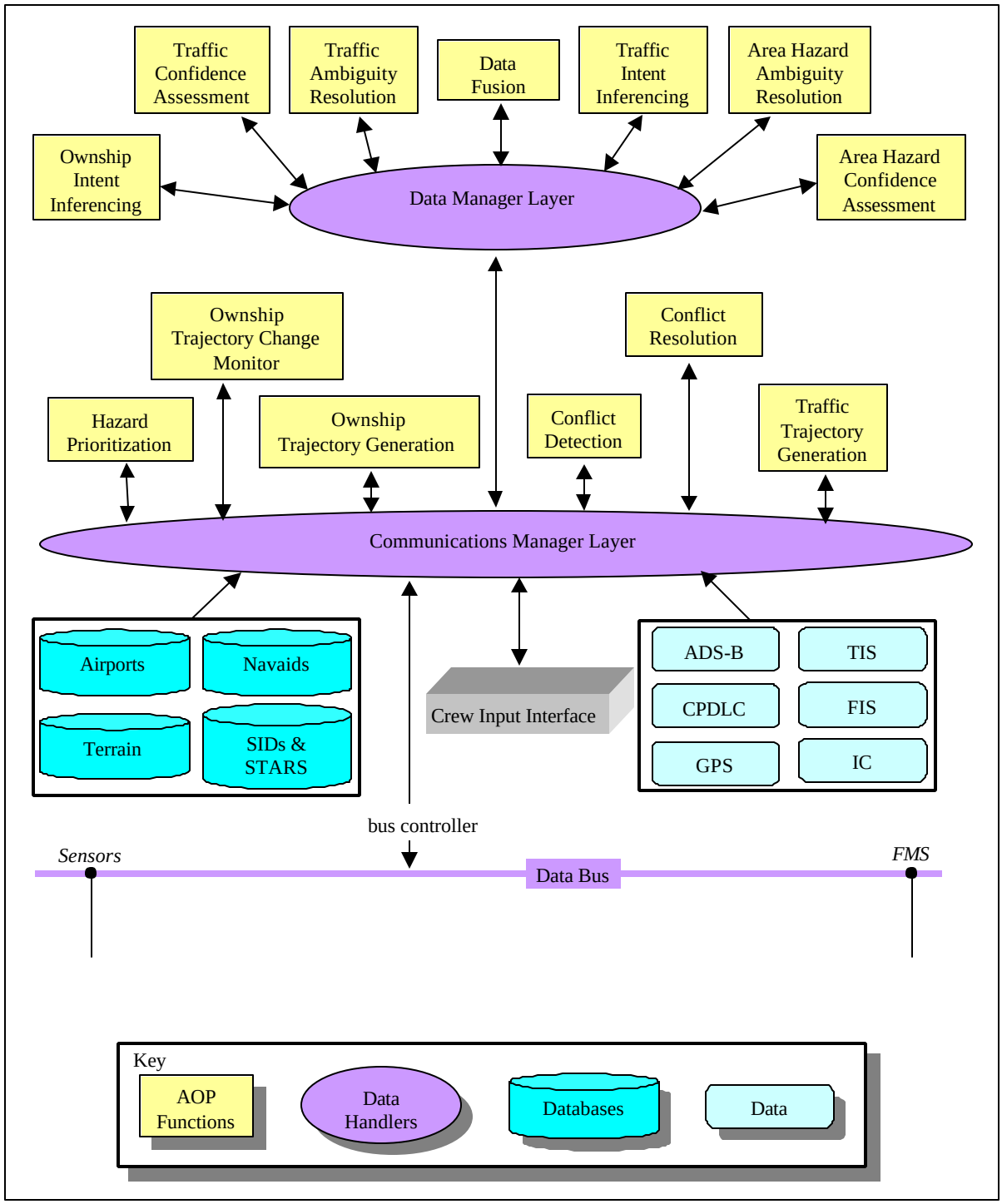


Figure 3-23: Proposed AOP Software Architecture

3.9.2.2.1 Data Bus

The data bus is the connection to the avionics of the aircraft. It contains both send and receive functions to deal with data from on-board sensors and the aircraft's FMS. In a simulation environment, the data bus may be modeled with software; however, it should maintain the same functionality of actual data bus hardware. Additionally, for simulation purposes, the data bus modeled for the AOP architecture does not have to be a specific standard (e.g.,

ARINC 629 or MIL-STD-1553), rather; a general functionality of a data bus may be sufficient. However, if the AOP is to be installed on a flight deck, then it is important to model the selected standard.

3.9.2.2.2 Communications Manager Layer

The Communications Manager (CM) manages the communication between all of the processes, some directly and some indirectly. CM is an event handler. It receives data and messages and sends them on to the intended recipient or recipients. It is designed to handle state changes, simulation flags, asynchronous and synchronous events. As shown in Figure 3-23, CM also is the connection to the databases and to all incoming data.

In this architecture CM is directly connected to Ownship and Traffic Trajectory Generators, the Ownship Trajectory Change Monitor, and the Hazard Prioritization, Conflict Detection and Conflict Resolution functions and to the Data Manger. These functions use the data that are handled by the Data Manager. CM is indirectly connected to the other AOP functions through the Data Manager. The Data Manager Layer is described in Section 3.9.2.2.3.

Upon receiving the IC message, CM passes them on so that the other functions may complete initialization. CM may monitor the health of other AOP processes and log data passing messages.

3.9.2.2.3 Data Manager Layer

The Data Manager Layer is in charge of storing all data and making the data available to all of the AOP functions, some directly and some through the Communications Manager. All information regarding the data is available to all of the AOP functions. Whenever any of the functions needs data to perform its task, the data are taken from a single source, the Data Manager. This architecture eliminates the passing of data directly from one function to another. The purpose is to maintain consistent data integrity and to eliminate the need for AOP functions to compute variables that have already been computed by other AOP functions.

For example, both the Intent Inference task and the Confidence Assessment task need data that come from multiple sources. Some of these data relate to the same thing. Data come from multiple sources at multiple rates. Weather data may come as 2D Grid-based data from one source and 3D Grid-based data from another source. There may also be Pilot Reports regarding local winds. The Data Fusion function combines those weather data that are related and come up with one model for the weather that is stored in the Data Manager along with the original data. On the other hand, traffic data may come from an ADS-B message and from a TIS message; however, the user may choose not to use the TIS data because they are of lower quality and lower frequency than the ADS-B data so these data sources will not be combined.

Another example is when multiple AOP functions rely on a common computation. For instance, the time to the point of closest approach or the point of closest approach itself are variables that are needed by several AOP functions. These variables should not be computed more than once, nor should they be computed by more than one definition of time to point of closest approach or point of closest approach. One AOP function should be responsible for computing the time to point of closest approach and point of closest approach data, and these values should be given to the CM for any other AOP function to use. In this way, all AOP functions use the same data that is computed only once without redundancy.

The AOP functions that are directly connected to the Data Manager in Figure 3-23 are those functions that manipulate the data.

4 Future Development

4.1 Conclusions

Over the course of the Fiscal Year 2000 effort, the redesign and enhancement of the FFSim Infrastructure software to satisfy the demands of the FFSim Phase 1R software architecture was successfully accomplished. New communication, surveillance, and simulation control models were specified including ADS-B, ATMFS, FIS-B, CPDLC, HCSE, Mode S data link, VDL-4 data link, and SSR. The critical components, ADS-B, ATMFS Executive, FIS-B, and CPDLC, were designed to be flexible and modular using object-oriented techniques. Then, these components were implemented in C++ software and successfully tested.

However, the full integration of the infrastructure software into the planned FFSim Phase 1R architecture (esp., ASTOR and SCC) remains to be completed. The major limiting factor in the integration to-date has been the absence of the design and development of the centralized SCC initialization, communication and control routines.

4.2 Recommendations for Future Work

Beyond the previously detailed software enhancement recommendations made in Sections 3.1.3, 3.2.3, 3.3.3, 3.4.3, 3.5.2, 3.6.2, and 3.7.2, there are a number of natural improvements to the developed FFSim infrastructure models that should be pursued. These recommendations are itemized in the following subsections.

4.2.1 Implement Communications Models

The Mode S and VDL-4 communication models should be implemented in software and applied to the ADS-B communications. Additionally, if resources exist, a UAT datalink communication model should be designed and implemented. Concurrently, we recommend that the output of the FAA SafeFlight21 program's Link Evaluation Team should be monitored in order to determine any persuasive arguments against any given ADS-B datalink implementation choice. If such persuasive arguments exist, the modeling of the insufficiently-performing datalinks should be curtailed.

Additional datalink communications modeling for CPDLC (i.e., VDL-3) and FIS-B (i.e., VDL-2 and UAT) could also be considered. However, the lower time criticality of the FIS-B information makes any datalink communications modeling supporting the FIS-B application not as important as that for ADS-B and CPDLC.

Where possible, NASA-Glenn's development of more detailed datalink communications models should be leveraged.

4.2.2 Expand CPDLC Message Set

Additional CPDLC messages as defined in [SW01] should be implemented as needed to support CE-5, CE-6, and CE-11 research.

4.2.3 Enhance Weather Simulation

Weather, while not a research area that is immediately driving near-term FFSim development, is a very important issue that will need to be simulated in order to determine feasibility of new Free-Flight-based operations. The current implementation of FIS-B is the first step, but more details of the real-world FIS-B system should be implemented in a future version of the simulation. Weather will be an important factor in both en route and terminal areas.

4.2.4 Implement Secondary Surveillance Radar

A Secondary Surveillance Radar model should be implemented. It is not recommended that Primary Surveillance Radar be implemented in the near future because of its inferior accuracy and diminishing role in the future NAS. The radar model in the current ADRS software should be compared with that designed under RTO-21 and a "best-of-breed" model should be implemented.

4.2.5 Integrate CTAS and PAS Software

Because of extensive efforts that have been conducted by NASA-Ames to ensure ongoing ADRS-CTAS and ADRS-PAS compatibility, we recommend that the ADRS software be integrated into the FFSim Phase 1R architecture. This should be done to avoid trying to re-engineer the detailed interfaces to these large, legacy software tools.

4.2.6 Reevaluate Implementation of TIS-B

Traffic Information Service – Broadcast (TIS-B) is not seen as a high-priority item to implement in FFSim. The major reason for this is because the accuracy is much less than ADS-B. Modeling TIS-B could provide some additional capability for simulating the period of transition to fully equipped aircraft, but even then its usefulness to an onboard decision support tool, e.g., AOP, is questionable. If it is necessary to determine the usefulness of TIS-B it is recommended that a simplified version be initially implemented. If the simplified version later proves TIS-B to be a useful FFSim component, a more extensive model could be implemented.

5 References

[A99] Anonymous, “Data Link Messages for TAP CTAS/FMS Experiment,” Updated June 25, 1999.

[AD01] http://adsb.tc.faa.gov/ADS-B/186-subf.htm

[C00] Conway, S., et al., “Autonomous Operations Planner Functional Specifications”, NASA Langley Research Center, Hampton, VA, Draft Version 5.0., December 2000.

[D01-1] Davis, P., “Free-Flight Simulation FFSim1R Automatic Dependent Surveillance – Broadcast (ADS-B) Software Design Version 1.0”, NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-10, Seagull Technology Inc., Los Gatos, CA, 2001.

[D01-2] Davis, P., “Free-Flight Simulation FFSim1R Flight Information Services – Broadcast (FIS-B) Software Design Version 1.0”, NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-11, Seagull Technology Inc., Los Gatos, CA, 2001.

[DE95] Denery, D. G. and Erzberger, H., “The Center-TRACON Automation System: Simulation and Field Testing,” *Proceedings of the advanced Workshop on ATM (ATM 95)*, sponsored by the National Research Council of Italy, Capri, Italy, Oct. 2-6, 1995. (Also published as NASA Technical Memorandum 110366, August, 1995).

[DKJ01] Dorsky, S., Krozel Ph.D., J., Jones, E., Schleicher, D., “Free-Flight Simulation FFSim1R Flight Information Services - Broadcast (FIS-B) Functional Specifications Version 1.0”, NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-05, Seagull Technology Inc., Los Gatos, CA, 2001.

[DMS01] Dorsky, S., Mueller, T., Schleicher, D., “Free-Flight Simulation FFSim1R Radar Functional Specifications Version 1.0”, NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-08, Seagull Technology Inc., Los Gatos, CA, 2001.

[DSD00] Davis, P., Schleicher, D., Dorsky, S., Wallace, E., Dow, D. Bjorkman, W., Miles, E., “Free Flight Simulation Infrastructure Fiscal Year 1999 Final Report”, NASA Prime Contract No. NAS2-98001, Technical Report 99175.21-01, Seagull Technology Inc., Los Gatos, CA, 2000.

[DW01] Davis, P., Wallace, E., “Free-Flight Simulation FFSim 1R Controller-Pilot Data Link Communications (CPDLC) Software Design Version 1.0”, NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-12, Seagull Technology Inc., Los Gatos, CA, 2001.

[DWS01] Davis, P., Wallace, E., Shah, S., Dow, D., Schleicher, D. “Free-Flight Simulation FFSim1R Air Traffic Management Facility Simulation Executive Functional Specifications Version 1.0”, NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-04, Seagull Technology Inc., Los Gatos, CA, 2001.

[EDG93] Erzberger, H., Davis, T. J., and Green, S., “Design of Center-TRACON Automation System,” *AGARD Guidance and Control Symposium on Machine Intellignce in Air Traffic Management*, Berlin, Germany, May, 1993.

[FAA98] *National Airspace System Architecture-1998*, Federal Aviation Administration, ASD-1, August 1998.

[GHJ95] Gamma, E., Helm, R., Johnson, R., Vlissides, J., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

[JBD97] Johnson, W.W., Battiste, V., Delzell, S., Holland, S., Belcher, S., Jordan, K., "Development and Demonstration of a Prototype Free Flight Cockpit Display of Traffic Information," *Proceedings of the 1997 SAE/AIAA World Aviation Congress*, 1997.

[KMS00] Krozel, J., Mueller, T., and Schleicher, D., "Intent Inference, Confidence Assessment, and Hazard Prioritization Status Report," NASA Prime Contract No. NAS2-98001, Technical Report 99175.30-02, Seagull Technology Inc., Los Gatos, CA, March 2000.

[LAS97] "LaSRS++ Programmer's Guide", Unisys Corporation, NASA Langley Research Center, Contract NAS1-20454, September 1997.

[LET99] SF21 ADS-B Link Evaluation Team, "Phase One Link Evaluation Report Status and Initial Findings", November 1999.

[MPW00] Communication with Jeff Maddalon, Mike Palmer, and David Wing, RTO-21 Design Review, NASA-Langley Research Center, Hampton, Virginia, August 2000.

[MS01-1] Mueller, T., Schleicher, D., "Free-Flight Simulation FFSim1R Mode S Communication System Model Functional Specifications Version 1.0", NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-07, Seagull Technology Inc., Los Gatos, CA, 2001.

[MS01-2] Mueller, T., Schleicher, D., "Free-Flight Simulation FFSim1R VHF Data Link - Mode 4 (VDL-4) Communication System Model Functional Specifications Version 1.0", NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-09, Seagull Technology Inc., Los Gatos, CA, 2001.

[PAS97] *Pseudo-Aircraft Systems (PAS) Users's Manual*, Revision 4.2, Logicon/Syscon/Syre, November, 1997.

[PAW97] Palmer, M. T., Abbott, T. A., Williams, D. H., "Development of Workstation-Based Flight Management Simulation Capabilities within NASA Langley's Flight Dynamics and Control Division", *Ninth International Symposium on Aviation Psychology*, Columbus, OH, April, 28-May, 1, 1997.

[RB00] Communication with Richard Barhydt, December 2000.

[RTCA95] *Report of the RTCA Board of Directors' Select Committee on Free Flight*, RTCA, January, 1995.

[RTCA195] RTCA Special Committee-195, *Minimum Aviation System Performance Standards (MASPS) for Flight Information Services – Broadcast (FIS-B) Data Link*, Draft Version 5.5, October 1999.

[RTCA219] *Minimum Operational Performance Standards for ATC Two-Way Data Link Communications*, RTCA, SC-169, DO-219, 1993.

[RTCA232] RTCA Special Committee-169, *Operations Concepts for Data Link Applications of Flight Information Services*, RTCA Document No. RTCA/DO-232, March 14, 1996

[RTCA242] RTCA Special Committee-186, *Minimum Aviation System Performance Standards for Automatic Dependent Surveillance Broadcast (ADS-B),* RTCA Document No. RTCA/DO-242, February 19, 1998.

[RTCA260] RTCA Special Committee-186, *Minimum Operational Performance Standards for 1090 MHz Automatic Dependent Surveillance Broadcast (ADS-B)*, RTCA Document No. RTCA/DO-260, September 13, 2000.

[SCD01] Shaw, S., Couluris, G.J., Davis, P., Dorsky, S., Schleicher, D., "Free Flight Simulation FFSim1R Host Computer System Emulator (HCSE) Functional Specifications Version1.0", NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-06, Seagull Technology Inc., Los Gatos, CA, 2001.

[SJD01] Schleicher, D., Jones, E., Davis, P., "Free-Flight Simulation FFSim1R Automatic Dependent Surveillance – Broadcast (ADS-B) Functional Specifications Version 1.0", NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-02, Seagull Technology Inc., Los Gatos, CA, 2001.

[SW01] Schleicher, D., Wallace, E., "Free Flight Simulation FFSim1R, Controller-Pilot Data Link Communications (CPDLC) Functional Specifications Version 1.0", NASA Prime Contract No. NAS2-98001, Technical Report 01175.21-03, Seagull Technology Inc., Los Gatos, CA, 2001.

Appendix A ADS-B Software Implementation Notes

As of the writing of this final report, the ASTOR software development was not in a sufficiently completed state to allow a full implementation of the FFSim ADS-B software. Therefore, a partial integration was performed, given the ASTOR development and RTO-21 resource and timing constraints. The following table (see Tables A-1 through A-5) documents the current state of the ADS-B integration with ASTOR, data element by data element.

Table A-1 Integration Status per ADS-B Data Element (1 of 5)

Values in DataManager	NASA's Originating Structure	Origin Data Variable	To Be Implemented in FFSim1R	Notes	Validity Flag Required	Currently Implemented
Aircraft Type	ASTOR/lib/DataMessages.lib (CduData)	TBD	yes	Originating Structure from SCC. Part of FMS info. MikePalmer Needs To Include in CduData. Does not exist in DO-242 ADS-B MASPS.	no	no, requires Mike Palmer to make CduData changes.
Barometric Altitude	ASTOR/lib/DataMessages.lib (SimData)	baroAltitude	yes		yes	yes
Barometric Altitude Rate	ASTOR/lib/DataMessages.lib (SimData)	verticalSpeed	yes		yes	yes
Beacon Code	TBD	TBD	no	Originating Structure from SCC init message	no	no
Calibrated Airspeed	ASTOR/lib/DataMessages.lib (SimData)	indicatedAirspeed	yes	If instrumentation and position error are negligible, indicatedAirspeed=Calibrated Airspeed. At this point, let's make that assumption. Does not exist in DO-242 ADS-B MASPS, but "Indicated Airspeed" does.	no	yes
Call Sign	ASTOR/lib/DataMessages.lib (CduData)	TBD	yes	Originating Structure from SCC. Part of FMS info. MikePalmer Needs To Include in CduData.	no	no, requires Mike Palmer to make CduData changes.
Category	ASTOR/lib/DataMessages.lib (CduData)	TBD	yes	Originating Structure from SCC. Part of FMS info. MikePalmer Needs To Include in CduData.	no	no, requires Mike Palmer to make CduData changes.
Cid	N/A	N/A	yes	Unique identifier (known as the Participant Address in DO-242) which will be specified by either the researcher or assigned by the SCC. This should not be a random # since you can't guarantee that a random number will be unique. The SCC needs to assign this number	no	yes, currently using random number created by transmitter.
East Velocity	ASTOR/lib/DataMessages.lib (SimData)	Calculated from Ground Speed and True Ground Track (Variable TBD)	yes	A function of groundspeed and track. Not currently in SimData however this value is calculated and expected to be included here. MikePalmer Needs To Include in CduData.	yes	no
Equipage Level	TBD	TBD	yes	Originating Structure from SCC init message	no	no
Geometric Altitude	XX	XX	no	Geo Alt and Geo Alt Rate will only matter when a non-standard atmosphere profile exists. Is not currently calculated by ASTOR/Sensor package (Source: Mike Palmer)	yes	no

Table A-2 Integration Status per ADS-B Data Element (2 of 5)

Values in DataManager	NASA's Originating Structure	Origin Data Variable	To Be Implemented in FFSim1R	Notes	Validity Flag Required	Currently Implemented
Geometric Altitude Rate	XX	XX	no	Is not currently calculated by ASTOR/Sensor package (Source: Mike Palmer)	yes	no
Geometric Position Valid	XX	XX	no	This MOPS data element refers to the validity of the latitude, longitude, and geometric altitude. In the ADS-B MOPS this consists of both "Geometric Position Valid-Horizontal" and "Geometric Position Valid-Vertical". This data element, except for reflecting extra bits of a real message element, will not provide lat/long/geo alt combined validity until FFSim3.	N/A – this is a validity flag	no
Ground Speed	ASTOR/lib/DataMessages.lib (SimData)	groundspeed	yes		yes	yes
Heading	ASTOR/lib/DataMessages.lib (SimData)	heading	yes	Refers to magnetic heading.	yes	yes
Indicated Airspeed	ASTOR/lib/DataMessages.lib (SimData)	indicatedAirspeed	yes		yes	yes
Latitude	ASTOR/lib/DataMessages.lib (SimData)	latitude	yes		yes	yes
Longitude	ASTOR/lib/DataMessages.lib (SimData)	longitude	yes		yes	yes
Mach	ASTOR/lib/DataMessages.lib (SimData)	mach	yes		yes	yes
Magnetic Ground Track	ASTOR/lib/DataMessages.lib (SimData)	track	yes	PcPlane provides Magnetic GroundTrack in degrees.	yes	yes
MCP Commanded Heading	TBD	TBD	no	To be replaced by "MCP Selected Heading"	yes	no
MCP Selected Heading	TBD	TBD	yes	Late entry into ADS-B requirements; therefore, basic integration info is still TBD; Software design documentation and code need to be updated to reflect terminology change from "MCP Commanded Heading" to "MCP Selected Heading"	yes	no
MCP Limit Altitude	TBD	TBD	no	To be replaced by "MCP Selected Altitude"	yes	no
MCP Selected Altitude	TBD	TBD	yes	Late entry into ADS-B requirements; therefore, basic integration info is still TBD; Software design documentation and code need to be updated to reflect terminology change from "MCP Limit Altitude" to "MCP Selected Altitude"	yes	no

Table A-3 Integration Status per ADS-B Data Element (3 of 5)

Values in DataManager	NASA's Originating Structure	Origin Data Variable	To Be Implemented in FFSim1R	Notes	Validity Flag Required	Currently Implemented
MCP Selected CAS/Mach	TBD	TBD	yes	Late entry into ADS-B requirements; need to add data element to ADS-B software design documentation and code	yes	no
MCP Selected Vertical Rate/Flight Path Angle	TBD	TBD	yes	Late entry into ADS-B requirements; need to add data element to ADS-B software design documentation and code	yes	no
North Velocity	ASTOR/lib/DataMessages.lib (SimData)	Calculated from Ground Speed and True Ground Track. (Variable TBD)	yes	A function of groundspeed and track. Not currently in SimData, however this value is calculated and expected to be included here. MikePalmer Needs To Include in CduData.	yes	no, requires Mike Palmer to make CduData changes.
Navigational Uncertainty Category - Position	TBD	TBD	no	To be split into "Navigational Accuracy Category – Position" and "Navigational Integrity Category – Position" as proposed in RTCA SC-186	no	no, requires SCC origin
Navigational Accuracy Category - Position	TBD	TBD	yes	Late entry into ADS-B requirements; Is not currently calculated by ASTOR/Sensor package (Source: Mike Palmer) For FFSim1R Originating Structure from SCC init message;	no	no, requires SCC origin
Navigational Integrity Category - Position	TBD	TBD	yes	Late entry into ADS-B requirements; Is not currently calculated by ASTOR/Sensor package (Source: Mike Palmer) For FFSim1R Originating Structure from SCC init message;	no	no, requires SCC origin
Navigational Uncertainty Category - Velocity	TBD	TBD	no	To be replaced by "Navigational Accuracy Category – Velocity"	no	no, requires SCC origin
Navigational Accuracy Category - Velocity	TBD	TBD	yes	Late entry into ADS-B requirements; Is not currently calculated by ASTOR/Sensor package (Source: Mike Palmer) For FFSim1R Originating Structure from SCC init message; Software design documentation and code need to be updated to reflect terminology change from "Navigational Uncertainty Category" to "Navigational Accuracy Category"	no	no, requires SCC origin
Surveillance Status	TBD	TBD	no	Used for indicating emergency/priority status	no	no
TCP Barometric Altitude	XnetMessages.lib (ProfileArrayXnet)	altitude	yes		yes	yes
TCP Calibrated Airspeed	XnetMessages.lib (ProfileArrayXnet)	cas	yes		yes	yes

Table A-4 Integration Status per ADS-B Data Element (4 of 5)

Values in DataManager	NASA's Originating Structure	Origin Data Variable	To Be Implemented in FFSim1R	Notes	Validity Flag Required	Currently Implemented
TCP Hour of ETA **TCP Minute of ETA** **TCP Second of ETA**	XnetMessages.lib (ProfileArrayXnet)	time	no	If this value is to be used to determine ETA a conversion is necessary from float to hr, min, sec plus current time. We need to know how time is defined.	yes	no
TCP Geometric Altitude		XX	no	Geometric Information not currently provided (Source: Mike Palmer)	yes	no
TCP Ground Speed	XnetMessages.lib (ProfileArrayXnet)	groundspeed	yes		yes	yes
TCP Heading	XX	XX	no	To be expunged from software design documentation and code per Richard Barhydt's direction; Not currently provided from FastWin (Source: Mike Palmer)	yes	no
TCP Indicated Airspeed	XnetMessages.lib (ProfileArrayXnet)	cas	yes	If instrumentation and position error are negligible, indicatedAirspeed=Calibrated Airspeed. At this point, let's make that assumption.	yes	yes
TCP Latitude **TCP Longitude**	XnetMessages.lib (ProfileArrayXnet.lib) ASTOR/lib/DataMessages.lib (CduData)		yes	This is a non-trivial calculation. For now it is assumed it will be done within the transmitter using information obtained from incoming messages. Need to use CduData route and profilearrayxnet information to calculate.	yes	no – need more detailed information on algorithm.
TCP Mach	XnetMessages.lib (ProfileArrayXnet)	cas and altitude	yes	Computed using algorithm pulled from FastWin.	yes	yes
TCP Magnetic Course From			no	Late entry into ADS-B requirements; SimData/track refers to route data and is the track angle between TCPx and TCPx+1, not the instantaneous value into and out of the point. TCP magnetic ground track information not currently available.	yes	no
TCP Magnetic Course To			no	Late entry into ADS-B requirements; SimData/track refers to route data and is the track angle between TCPx and TCPx+1, not the instantaneous value into and out of the point. TCP magnetic ground track information not currently available.	yes	no

Table A-5 Integration Status per ADS-B Data Element (5 of 5)

Values in DataManager	NASA's Originating Structure	Origin Data Variable	To Be Implemented in FFSim1R	Notes	Validity Flag Required	Currently Implemented
TCP Magnetic Ground Track Angle			no	In conjunction with "TCP True Ground Track Angle", to be replaced by "TCP Magnetic Course From" and "TCP Magnetic Course To";	yes	no
TCP Hour of Time to Go **TCP Minute of Time to Go** **TCP Second of Time to Go**	XX	XX	no	Not currently provided from FastWin (Source: Mike Palmer)	yes	no
TCP Name			no	A waypoint name	no	no
TCP True Air Speed			no	Can be computed. Need access to either temperature deviation or static air temperature for algorithms.	yes	no
TCP True Ground Track Angle	XX	XX	no	In conjunction with "TCP Magnetic Ground Track Angle", to be replaced by "TCP Magnetic Course From" and "TCP Magnetic Course To"; Not currently provided from FastWin (Source: Mike Palmer). However, could be calculated from Magnetic Ground Track (when available)	yes	no
TCP Type	XnetMessages.lib	flag	no	i.e., waypoint, TOC, TOD. Contact Dave Williams for more detailed information	no	no
True Airspeed	ASTOR/lib/DataMessages.lib (SimData)	trueAirspeed	yes		yes	yes
True Ground Track	ASTOR/lib/DataMessages.lib (SimData)	track + magneticVariation	yes	Using track and magneticVariation from the SimData structure, true ground track can be calculated. Ultimately this should be calculated in the Sensor Package.	yes	yes
Turn Indication	ASTOR/lib/DataMessages.lib (SimData)	turnDirection	yes		no	yes
Utc Day	ASTOR/lib/DataMessages.lib (SimData)	utcDay	yes		no	yes
Utc Hour	ASTOR/lib/DataMessages.lib (SimData)	utcHours	yes		no	yes
Utc Minute	ASTOR/lib/DataMessages.lib (SimData)	utcMinutes	yes		no	yes
Utc Month	ASTOR/lib/DataMessages.lib (SimData)	utcMonth	yes		no	yes
Utc Seconds	ASTOR/lib/DataMessages.lib (SimData)	utcSeconds	yes		no	yes